Министерство науки и высшего образования
Российской Федерации

Национальный исследовательский ядерный университет «МИФИ»

А. В. Кузнецов

Основы LaTeX

Учебное пособие

Издание второе, исправленное и дополненное

Москва 2024

УДК 681.322
ББК 32.97
К 89

Кузнецов А.В. **Основы LaTeX.** *Учебное пособие.* Изд. 2-е, испр. и доп. М.: НИЯУ МИФИ, 2024. 504 с.

Книга, посвященная верстке в издательской системе LaTeX, ориентирована как на начинающих авторов, так и на тех, для кого LaTeX — привычный инструмент. В ней описаны ресурсы стандартного LaTeX, достаточные для оформления дипломов, диссертаций, рукописей научных статей и книг.

В основу изложения положена логика работы компиляторов, выполняющих различные операции. Рассмотрены основные приемы верстки текста и формул, создания рисунков и таблиц; формирование библиографии, оглавления и предметного указателя.

Книга содержит большой объем справочной информации, собранной в таблицах, и обширный указатель.

Издание предназначено для студентов и аспирантов физико-математических специальностей, а также авторов, самостоятельно верстающих тексты.

Рецензенты:
д-р физ.-мат. наук, проф. С.В. Попруженко
(кафедра теоретической ядерной физики НИЯУ МИФИ);
канд. физ.-мат. наук, доц. А.А. Никитин
(факультет вычислительной математики и кибернетики
МГУ им. М.В. Ломоносова).

ISBN 978-5-7262-3015-3

Предисловие

В настоящее время нет нужды доказывать преимущества издательской системы LaTeX в оформлении статей и книг научно-технической и особенно физико-математической тематики. Она прочно заняла ведущие позиции во всех издательствах, специализирующихся на выпуске научной литературы и периодики, так как сводит к минимуму редакционную обработку рукописей и существенно сокращает сроки их опубликования. В ее основе лежит концепция программной верстки, заложенная Дональдом Е. Кнутом, создавшим в 1978 г. язык программирования и компилятор TeX, предназначенные для верстки текста, насыщенного математическими формулами [1]. Создавая документ, автор пишет программу, команды которой указывают, что нужно сделать с той или иной частью текста. Выполняя данную программу, компилятор TeX верстает документ.

В 1984 г. Лэсли Лэмпорт разработал на основе TeX удобный и компактный макроязык LaTeX, кардинально минимизировавший количество команд разметки, необходимых для форматирования рукописи. Он обогатил концепцию программной верстки понятием стиля, или класса документа. Размечая текст, автор использует достаточно небольшой набор команд, имеющих различные настройки. Настройки помещаются в стилевой файл, определяющий вид сверстанного документа. Изменение стиля верстки не требует изменения текста программы, за исключением команды, загружающей нужный стиль.

Введенные дополнения оказались очень эффективными. К началу 90-х гг. ведущие издательства научной литературы разработали собственные стили, отвечающие их полиграфическим традициям, и ввели LaTeX в издательский процесс. Появилось большое количество пакетов, т.е. дополнительного программного обеспечения, существенно расширивших возможности LaTeX. С годами их число неуклонно растет. Так как TeX и LaTeX развиваются и распространяются усилиями сообщества свободного программирования, любой пользователь имеет право разработать новый пакет, решающий какую-то задачу верстки, зарегистрировать его и присовокупить к дистрибутиву LaTeX, насчитывающему в настоящее время более трех тысяч пакетов. Как правило, такие пакеты не конфликтуют с ядром LaTeX, однако могут конфликтовать между собой. С этой точки зрения все программное обеспечение следует разделить на стандартный LaTeX, состоящий из ядра и небольшого числа пакетов, поддерживаемых и развиваемых

основной командой разработчиков, и нестандартные пакеты, разработанные и поддерживаемые многочисленными энтузиастами. В практике выпуска научной периодики издательства используют стандартный LaTeX, остальные ресурсы могут отсекаться на стадии подачи рукописи в редакцию, поэтому нестандартные пакеты можно использовать лишь на свой страх и риск в документах, не предназначенных для публикации. Начиная с 1993 г. стандартом является версия LaTeX 2_ε, ставшая за прошедшие годы очень стабильной. Именно она описана в данной книге. Много лет ведется разработка LaTeX 3, направленная на полную совместимость с форматом unicode, гибкую работу со шрифтами, интеграцию ряда пакетов непосредственно в ядро LaTeX и расширение возможностей компилятора за счет использования скриптов языка Lua. Одна из основных задач проекта — расширение возможности основных команд за счет введения дополнительных настроек без изменения их интерфейса, чтобы при переходе к новому стандарту сохранить возможность использовать ранее созданные документы. Многие наработки незаметно для пользователей уже внесены в LaTeX 2_ε.

В литературе [2–10] описано большое число пакетов, используемых для оптимизации различных аспектов верстки. Однако в подавляющем числе случаев автору, работающему над рукописью, вполне достаточно средств стандартного LaTeX. Ему не нужно погружаться в изучение богатых дополнительных ресурсов. Его интересует не процесс или оптимизация верстки, а результат, достигаемый наиболее коротким путем. Данное руководство предназначено служить подспрорьем таким пользователям. Оно не претендует ни на тщательность, с которой создание документа описано в книге С. М. Львовского «Набор и верстка в пакете LaTeX» [2], ни на полноту описания стандартного LaTeX в книге И. Котельникова и П. Чеботаева «LaTeX 2_ε по-русски» [3], ни на фундаментальность обзора основных и дополнительных средств LaTeX в серии книг М. Гуссенса, Ф. Миттельбаха, С. Ратца и А. Самарина [4–6]. Опираясь на опыт преподавания курса LaTeX студентам Московского инженерно-физического института и общение с коллегами, активно использующими LaTeX в своей практике, автор стремился свести к разумному минимуму набор обсуждаемых средств, считая его вполне достаточным для работы над статьей, дипломом и даже книгой. Насколько это удалось, судить Вам, читатель.

В книге изложены основы стандартного LaTeX, позволяющие создавать и редактировать документы. Изложение разбито на две части, преследующие разные цели.

В первой описаны основные принципы верстки текста и формул, показана работа с графикой, приведены методы создания таблиц и списка литературы. Изложение строится не на описании рецептов, как выполнить ту или иную операцию, а на логике работы компилятора, поняв которую, автор сможет найти оптимальное решение сам. Освоения материала первой части достаточно для создания документов, не требующих вмешательства в алгоритмы автоматической верстки.

Вторая часть адресована авторам, желающим освоить LaTeX более глубоко. В ней обсуждаются дополнительные возможности, имеющиеся в базовом LaTeX и предоставляемые стандартными пакетами, круг которых ограничен лишь теми, что используют в своей практике ведущие издательства технической литературы. Описанные ресурсы позволяют настраивать различные аспекты верстки, а также создавать большие документы и книги.

Автор выражает глубокую признательность А.И. Маймистову, А.А. Иванову и А.А. Синченко за полезные рекомендации и ценные замечания к рукописи.

А.В. Кузнецов

Предисловие ко второму изданию

Второе издание книги расширено обсуждением современных шрифтовых технологий, используемых компиляторами LuaTeX и XeTeX. Хотя они пока не являются стандартом в издательствах, выпускающих научную периодику, но претендуют стать им в ближайшем будущем. Описанные средства позволяют верстать документы OpenType и TrueType шрифтами, большое количество которых содержат различные операционные системы и дистрибутивы TeX. В настоящее издание добавлен обзор возможностей пакетов fontspec и unicode-math, обеспечивающих эффективную работу с этими шрифтами.

Ресурсы верстки формул, представленные в первом издании, были ограничены лишь теми, что поставляет ядро LaTeX и пакеты коллекции AMS-LaTeX. В настоящее время они совершенствуются, но не расширяются. Деятельность по разработке новых конструкций перенесена в пакет mathtools, описание которого включено в настоящее издание.

Часто возникает необходимость оформления документации в соответствии с действующими ГОСТами. Их требования должны выполняться при подготовке отчетов о НИР (ГОСТ 7.32-2017 «Отчет о научно-исследовательской работе») и верстке диссертаций (ГОСТ Р 7.0.11-2011 «Диссертация и автореферат диссертации»). В настоящем издании обсуждается настройка стандартных средств LaTeX для выполнения этой задачи.

В книге упоминаются ресурсы большого количества пакетов. В зависимости от целесообразности их применения они описаны с различной полнотой, а часть из них и вовсе осталась «за кадром» нашего рассмотрения. Обсуждение всех ресурсов сильно увеличило бы объем книги и загромоздило изложение ненужными, с точки зрения автора, деталями. Многие пакеты продолжают развиваться, их возможности расширяются, поэтому в списке литературы приведены интернет-адреса с документацией пакетов. Автор рекомендует обращаться к ней, если описанных ресурсов окажется недостаточно, или они начнут действовать несколько иначе, чем сказано в книге.

Ошибки, неточности и опечатки, замеченные в тексте первого издания, исправлены, но, возможно, добавлены и новые. Автор выражает признательность В.С. Рыкованову за большую помощь в первой части этой работы, а вторая полностью остается на моей совести.

А.В. Кузнецов

Принятые обозначения

Документ LaTeX представляет собой программу, выполняя которую, компилятор TeX формирует документ по заданному шаблону. Чтобы избежать путаницы, формируемый при компиляции файл мы будем называть *документом*, а текст, редактируемый автором, — *рукописью*. Наиболее часто для представления документов используются форматы PostScript и PDF, обеспечивающие высокое полиграфическое качество и адекватность воспроизведения в любой операционной системе (ОС).

Книга содержит много примеров, поясняющих обсуждаемый материал. Короткие примеры представляют собой конструкцию типа «`часть рукописи` ⇝ часть документа», показывающую, как будет выглядеть часть документа после компиляции `части рукописи`. Они приводятся непосредственно в тексте, например `$\sin x$` ⇝ $\sin x$. Громоздкие примеры выносятся в отдельные абзацы. В них символ «⇝» опускается:

```
$$ \delta_{ij} =
     \begin{cases}
        1, & i=j,\\
        0, & i\ne j.
     \end{cases}  $$
```

$$\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

В некоторых простых случаях, чтобы избежать дублирование, код, набранный мелким шрифтом и подчеркнутый пунктиром, печатается вместе с результатом его действия, как в следующем предложении. В примерах код рукописи печатается `\texttt{шрифтом «пишущей машинки»}`.

Информация, не имеющая определенного значения или несущественная в данном контексте, заменяется символом «•». Например, он заменяет имена файлов с расширением tex, обозначаемых как •.tex. В предметном указателе с помощью этого символа показана структура команд и окружений, имеющих параметры и аргументы, `\rule[•]{•}{•}` и table`[•]`, а также наличие настроек у пакетов, например graphicx`[•]`.

Команды и другие объекты, на которые ссылается предметный указатель, для облегчения поиска выделены серым фоном.

Настройки команд, окружений и пакетов обычно содержат какие-либо значения, используемые по умолчанию. В списках настроек такие значения выделяются жирным шрифтом, а варианты настроек разделяются символами «|», например:

landscape | **portrait** — ориентация страниц.

Часть I

Верстка в LaTeX

Глава 1

Создание документа

Для работы в LaTeX необходимы компиляторы, набор программных средств — *пакетов* в терминологии LaTeX, и графическая оболочка, обеспечивающая комфортное редактирование рукописи, а также управление компиляторами и средства просмотра готовых документов. Вдобавок к этому для формирования списков литературы желательно иметь средства работы с библиографическими базами.

В настоящее время во всех операционных системах имеется одинаковый набор средств работы в LaTeX. Освоив их, автор может использовать любую ОС, не испытывая неудобства работы с незнакомым программным обеспечением.

Рукопись можно редактировать в обычном текстовом редакторе, но намного удобней пользоваться специализированными программами имеющимися во всех ОС. Лучшей графической оболочкой является программа TeXstudio, имеющая собственные средства просмотра pdf-документов. Ресурсы LaTeX содержит дистрибутив MiKTeX, устанавливающий базовый набор компиляторов и программных средств, необходимых для верстки. Дистрибутив TeXLive, развивающийся ОС Linux, содержит самые свежие разработки компиляторов и программных средств. Удобный интерфейс работы с библиографическими базами предоставляет написанная на языке Java программа JabRef.

Все перечисленные программы распространяются и используются свободно. Они активно развиваются и потому описывать их интерфейс — задача неблагодарная, если не бессмысленная. Однако автор счел необходимым сделать это, чтобы читатели могли начать самостоя-

тельную работу в LaTeX. К счастью, данные программы разрабатываются уже не первое десятилетие, их интерфейс уже в основном сложился и вряд ли будет кардинально меняться в ближайшем будущем.

1.1. Установка и настройка программ

MiKTeX можно установить непосредственно из интернета в полном или базовом варианте. Полный дистрибутив занимает несколько гигабайтов, при этом для работы нужна лишь малая его часть. Базовый комплект содержит только основные программные средства, а остальные добавляются по мере надобности. По умолчанию недостающие компоненты загружаются из интернета, что не всегда удобно, так как их получение зависит от доступности сети. В то же время набор необходимых средств формируется довольно быстро, и после непродолжительной работы острая потребность в интернете пропадает.

Стандартным путем является загрузка с сайта `http://MikTeX.org/` и запуск программы basic-MikTeX-•-x#.exe, устанавливающей базовый комплект MiKTeX. Здесь • обозначает текущую версию MiKTeX, а # — разрядность компьютера. Запустите программу и выберите вариант установки «Install MikTeX only for me». В этом случае дистрибутив установится в папки, к которым вы будете иметь полный доступ, а для добавления пакетов не потребуются права администратора. После установки запустите программу miktex-console, панель которой показана на рис. А.1, войдите в меню «Updates» и обновите установленные программы и пакеты. Выполнив данные действия вы получите базовый комплект MiKTeX, который по мере необходимости будет автоматически устанавливать недостающие пакеты из интернета. Его следует подстроить, добавив шрифты для генерации PostScript- и pdf-документов.[1] Для этого вновь запустите программу MikTeX Console, откройте меню «Packages», в строке поиска введите название пакета cm-super, нажмите правой клавишей мыши на появившуюся строку с его описанием и в выпавшем меню выберите «Install package». После этого MiKTeX полностью готов к работе. Более подробно его устройство и настройка описаны в прил. А.

[1] По умолчанию во время компиляции «на лету» генерируются растровые (пиксельные) шрифты. Их качество хуже, чем у векторных шрифтов, поддерживаемых форматами PostScript и pdf. Установка векторных шрифтов cm-super устраняет этот недостаток.

Вдобавок к MiKTEX нужно установить программу TeXstudio, загрузив ее с сайта `https://texstudio.org`. В ходе установки она автоматически настроит взаимодействие с MiKTEX, поэтому ее следует устанавливать за ним. Установив ее, загрузите русский словарь, используемый для проверки орфографии набираемого текста. Для этого запустите TEXstudio и в меню «Options ⇒ Configure TeXstudio» выберите «Language Checking», открыв ссылку «LibreOffice», найдите и скачайте последнюю версию русского словаря, а затем, нажав кнопку «ImportDicionary...», установите его. Если вы преимущественно работаете с русскими текстами, имеет смысл загружать его при запуске TEXstudio. Это можно сделать, выбрав его в меню «Default Language».

Установив MiKTEX и TEXstudio, вы получите полный набор средств для работы с LATEX. Чтобы сделать работу комфортной, рекомендуем поближе познакомиться с интерфейсом графической оболочки и подстроить ее под свои потребности. Остановимся кратко на ее ресурсах, необходимых в первую очередь.

Рабочее окно программы TEXstudio показано на рис. 1.1. В его левой части находится многофункциональная панель. В центральной части расположено окно редактора, а под ним окно, в которое выводятся сообщения компиляторов. Справа находится окно, показывающее сверстанный документ. В нижнем левом углу рабочего окна есть иконки, позволяющие свернуть или открыть многофункциональную панель и окно сообщений.

Многофункциональной панелью управляют иконки, расположенные слева от нее. По умолчанию в ней активируется навигационная панель (верхняя иконка), показывающая структуру загруженных файлов и содержащиеся в них метки. Выбрав метку или раздел, можно переключиться в соответствующую позицию файла.

В панели, активируемой второй иконкой сверху, открывается оглавление документа, формируемое при компиляции. Выбранный раздел оглавления открывается в окне редактора.

В еще одной панели (третья иконка) навигация осуществляется по закладкам, которые можно создать в тексте.

Иконка с изображением звездочки активирует показанное на рис.1.2 графическое меню ввода символов, использование которого ограждает от опечаток и обусловленных ими ошибок компиляции. Справа от строки поиска в нем находится выпадающий список, позволяющий переключаться между различными наборами символов. По умолчанию данное меню вставляет в текст рукописи команды, но его можно настроить и на

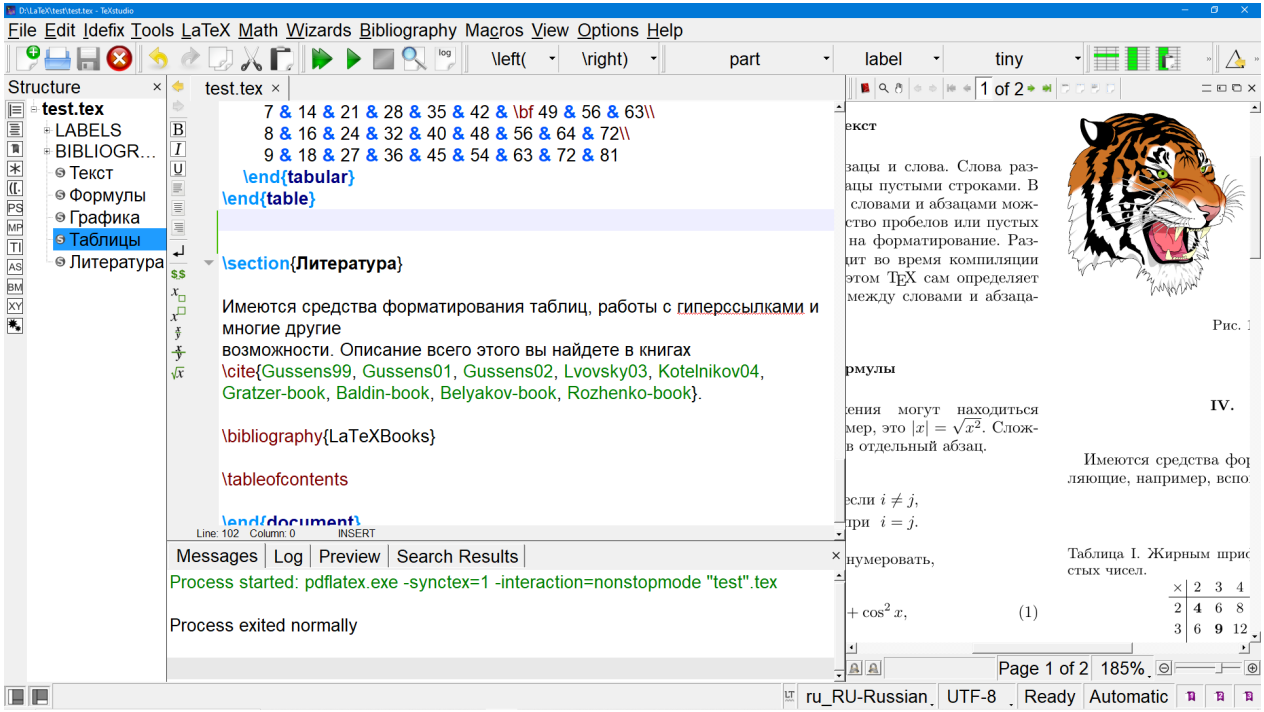


Рис. 1.1. Рабочее окно программы TEXstudio

печать символов. Для этого откройте меню «Options ⇒ Configure TeXstudio ⇒ Adv. Editor» и активируйте флажок «Insert Symbol as Unicode». Символы, отсутствующие в графическом меню, можно напечатать, используя меню «Edit ⇒ Insert Unicode Character...». Оно открывает окно для ввода номера символа в кодировке unicode. Введенный символ демонстрируется в окне, а при его закрытии вставляется в рукопись.

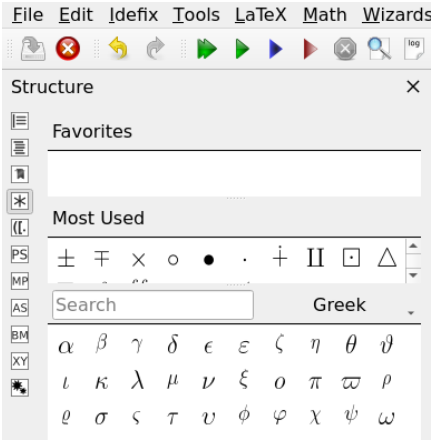


Рис. 1.2. Меню ввода символов программы TEXstudio

Остальные ресурсы многофункциональной панели обслуживают конструкцию `\left•...\right•` (см. разд. 4.1.4) и команды макропакетов, предназначенных для работы с графикой, построения сложных диаграмм и создания презентаций.

Рабочее окно содержит меню «LaTeX» и «Math»(см. рис. 1.1), в которых находятся средства верстки текста и формул. Меню «Wizards» позволяет создать основу нового документа, сделать заготовку таблицы и вставить иллюстрацию.

В меню «Options ⇒ Configure TeXstudio» находится окно настроек графической оболочки. Чтобы получить доступ ко всем параметрам настройки, активируйте флажок «Show Advanced Options» в его левом нижнем углу. Обширный набор настроек хорошо структурирован, поэтому отыскать интересующую настройку, если она предусмотрена, можно довольно легко. Меню «General» позволяет установить шрифт интерфейса TEXstudio, а в меню «Editor» можно выбрать шрифт окна редактора.

По умолчанию, при наборе открывающей скобки редактор автоматически добавляет к ней закрывающую, а если выделен фрагмент текста, заключает его в скобки. Сняв флажок «Auto Complete Parenthesis» в меню «Adv. Editor» окна настроек, можно изменить режим ввода на традиционное редактирование, при котором скобки вводятся как обычные символы.

1.2. Загрузка и сохранение файлов

TEXstudio, по умолчанию, настроен на работу с файлами в кодировке UTF-8, которую без особой нужды лучше не менять, так как TEXstudio использует ее, редактируя рукопись. Файлы, имеющие другие кодировки, при загрузке перекодируются в UTF-8, а при сохранении возвращаются в исходную. На обоих этапах могут возникнуть неприятности.

Если загруженный текст превращается в абракадабру, измените кодировку загрузки и повторите загрузку. Для этого в меню кодировок в нижней строке окна редактора или меню «Edit ⇒ Setup Encoding» выберите нужную кодировку[2] и нажмите клавишу «Reload With».

После того как текст станет читаемым, снова откройте меню, выберите кодировку UTF-8 и нажмите клавишу «Change To», чтобы файл сохранялся в кодировке UTF-8. Если это не сделать, кодировка загрузки будет использована и при сохранении.

Если абракадабру содержит сверстанный документ, проверьте кодировку текста, объявленную в команде

```
\usepackage[кодировка]{inputenc}
```

(см. разд. 2.6.4), она должна соответствовать кодировке файла.

1.3. Компиляция и просмотр документа

Организуя процесс компиляции, TEXstudio записывает рукопись в файл и указывает компилятору путь к нему. Таким образом, внесенные в текст изменения автоматически сохраняются перед компиляцией.

В подготовке документа участвует несколько программ, каждая из которых делает свою часть работы. При работе они генерируют ряд

[2] Обычно проблема возникает при загрузке файлов в кодировке windows-cp1251, поэтому, прежде всего, следует опробовать ее.

файлов и посредством них обмениваются информацией. Для успешного выполнения своей функции каждая из них нуждается в данных, подготовленных другими программами. Чтобы сверстать документ в окончательном виде, зачастую нужно последовательно запустить несколько программ в строго определенном порядке. TEXstudio формирует цепочку компиляции, запускает ее и следит за сообщениями компиляторов во время их работы.

Блок-схема процесса компиляции представлена на рис. 1.3. Генерируемые файлы наследуют имя компилируемого документа, названного в нашем примере «document», а их расширение определяет заключенная в них информация. Серым цветом на блок-схеме выделены файлы, загружаемые как внешние ресурсы, которые компиляторы не изменяют.

Основным компилятором является программа pdftex. Для ее запуска используется программа pdflatex, передающая компилятору требуемые настройки. При компиляции в качестве внешних ресурсов помимо текста рукописи и рисунков загружаются следующие файлы: форматный файл LATEX (•.fmt), класс документа (•.cls) и пакеты (•.sty) и (•.tex). В скобках указаны расширения имен файлов. Кроме этого загружаются файлы, созданные в ходе предшествующих компиляций: оглавление (•.toc), предметный указатель (•.ind), библиография (•.bbl) и информация, необходимая для нумерования разделов, формул, цитируемых источников и т. д. (•.aux).

В процессе работы компилятор TEX создает документ (•.pdf или •.dvi) и генерирует ряд служебных файлов, содержащих сведения для предметного указателя (•.idx), оглавление (•.toc) и отчет о компиляции (•.log). В файл с расширением •.aux заносится информация о порядке следования рисунков, таблиц, цитируемых источников и других объектов в тексте рукописи.

Если документ сохраняется в формате dvi, программа dvips конвертирует его в PostScript.

Предметный указатель составляют программы MakeIndex, xindy или xindex. Из файла с расширением •.idx они считывают список терминов, формируют указатель и сохраняют его в файле с расширением •.ind, записав отчет о компиляции в файл с расширением •.blg.

Программа bibtex создает библиографию. Из файла с расширением •.aux она считывает список цитируемых источников, загружает библиографические базы (•.bib) и стиль оформления библиографии (•.bst), формирует список литературы, сохраняет его в файл с расширением •.bbl и направляет в файл с расширением •.blg отчет о компиляции.

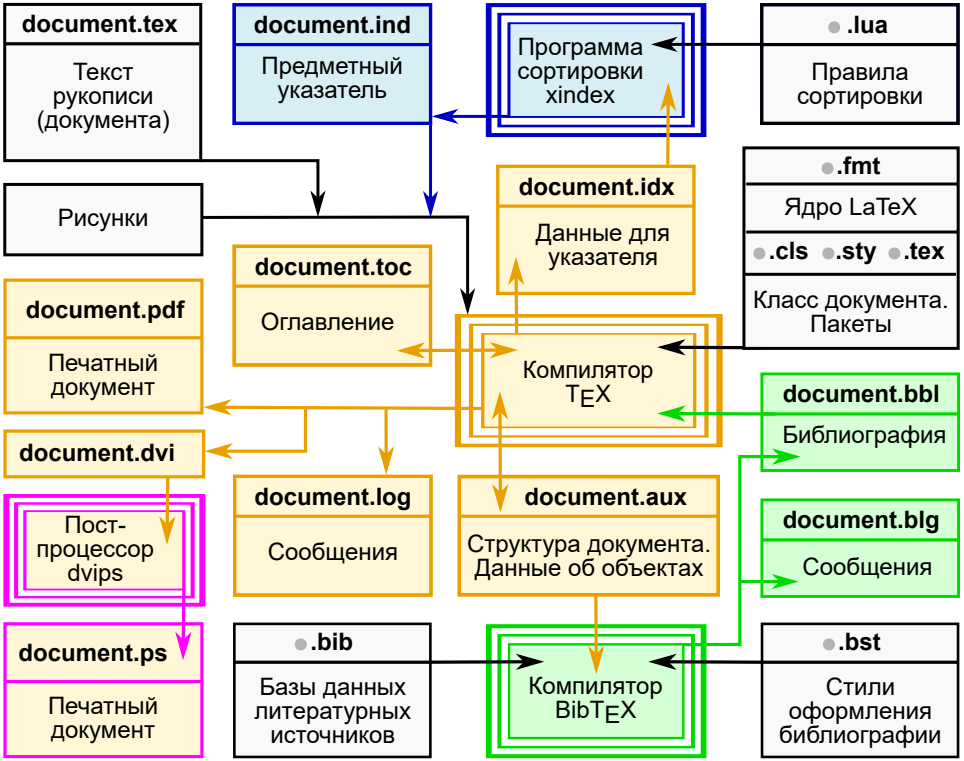


Рис. 1.3. Блок-схема процесса компиляции

Таким образом, помимо самого документа компиляторы генерируют значительное число дополнительных файлов, поэтому для каждого нового документа рекомендуется создать собственную рабочую папку. Файлы, генерируемые в процессе компиляции, наследуют имя файла с текстом рукописи, которое помещается в команду `\jobname`, используемую для генерации других имен, например `\jobname.pdf`.

Файлы с расширениями •.aux, •.toc, •.lof, •.lot, •.bbl и •.ind используются в последующих компиляциях. При смене класса документа или изменении набора загружаемых пакетов они являются потенциальным источником ошибок, так как содержащиеся в них команды могут стать неопределенными, или начнут конфликтовать с другими командами. В таком случае файлы с данными расширениями следует удалить и начать компиляцию «с чистого листа». В TeXstudio для этого предусмотрено меню «Tools ⇒ Clean Auxiliary Files». Меню «Tools» содержит также средства запуска различных комбинаций компиляции, описанные в прил. А. На панели задач имеются иконка ▶, запускающая компилятор LaTeX, и иконка ⏩, инициирующая цепочку компиляций, формирующую все компоненты документа, включая библиографию и предметный указатель.

TeXstudio имеет окно просмотра pdf-документов, расположенное справа от окна редактора (см. рис. 1.1). После компиляции в нем показывается страница, содержащая абзац, в котором находится курсор окна редактора.

Окна редактирования и просмотра между собой связаны. Наведя курсор и нажав клавишу «Control» вместе с левой клавишей мыши, можно переключиться в соответствующее место рукописи или документа. Тот же переход можно совершить, нажав правую клавишу мыши и выбрав «Go to Source» или «Go to PDF» в появляющемся меню.

В ОС Windows для просмотра компилируемых документов не следует пользоваться программой AdobeReader. Открывая файл, она блокирует его изменение, что вызывает сбой работы компиляторов, которые не могут сохранить сверстанный документ.

1.4. Основные понятия и синтаксис LaTeX

Программа LaTeX состоит из текста, команд и комментариев. Строго говоря, имеются еще и формулы, которые также состоят из текста, т. е. символов, обозначающих переменные, команд и комментариев и потому, с точки зрения синтаксиса, не являются особыми объектами.

Текст разбивается на слова, разделяемые пробелами, и абзацы, разделяемые пустыми строками. Между словами можно поставить любое количество пробелов, а между абзацами любое количество пустых строк, при компиляции они игнорируются. В непустых строках символы перевода строки считаются обычными пробелами.

Верстая документ, компилятор сам разбивает его текст на строки и устанавливает промежутки между словами и строками, исходя из заданного стиля верстки. Дополнительные пробелы следует использовать для наглядного представления структуры рукописи и конструкций LaTeX. Сравните, например, два варианта кода, один из которых не очень удобен для восприятия:

```
$$\delta_{ij}=\begin{cases}1,&i=j\\0,&i\ne j\end{cases}$$,
```

а второй — прост и понятен:

```
$$ \delta_{ij} =
     \begin{cases}
        1, & i=j,\\
        0, & i\ne j.
     \end{cases}  $$
```

$$\delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \ne j. \end{cases}$$

Оба они после компиляции дают одинаковый результат.

Комментарий открывается символом процента `%` и ограничивается концом строки. В комментариях, состоящих из нескольких строк, каждая из них должна начинаться символом процента. Текст и команды, находящиеся в комментарии, при компиляции полностью игнорируются. Комментарии могут находиться в любом месте рукописи, в том числе в аргументах и параметрах команд. Комментарий не является пробелом и потому не разрывает текст, но попытка разбить комментарием имя команды вызовет ошибку. Проиллюстрируем сказанное следующим примером:

```
Текст %    A
будет не%  комментарий
прерывен!% пропадет!
```

Текст будет непрерывен!

1.4.1. Команды и окружения

Признаком команды является обратный слэш, за которым следует ее имя, параметры и аргументы, например

`\samplecommand`[*параметр*]{*аргумент*}.

Имя команды состоит из латинских букв и звездочки `*`, которая может стоять в конце него. Аргументы команд заключаются в фигурные

скобки, а параметры в квадратные. Прописные и строчные буквы в именах различаются, поэтому, например, команды `\AA` ⇝ Å и `\aa` ⇝ å производят разные символы.

LaTeX не имеет какого-либо специального признака окончания имени команды. Им служит любой символ, отличный от букв и звездочки. После имени с равным успехом могут стоять как пробел, так и скобки, знаки препинания цифры и т. д., в любом случае компилятор распознает команду.

Строение команд не ограничено каким-либо шаблоном. Наличие и количество аргументов и параметров вводится при их определении. LaTeX не регламентирует порядок следования аргументов и параметров, однако есть общие правила, которым они подчиняются. Во-первых, аргументы и параметры всегда следуют *за* именем команды. Во-вторых, они могут содержать другие команды, пробелы, комментарии и т. д. И наконец, между именем, аргументами и параметрами допускаются пробелы и комментарии, но не должно быть пустых строк.

Аргумент содержит информацию, без которой команда не может быть выполнена, или объекты, с которыми она должна совершить какое-то действие. Если аргумент не выделен явно фигурными скобками, вместо него используется символ (или команда), следующий непосредственно за командой, например, `\textbf\AA` ⇝ **Å**. Если полученный таким образом аргумент не удовлетворяет требованиям команды, компилятор выдаст сообщение об ошибке. В частности, аргументом команды `\hspace{`*длина*`}` служит *длина*, характеризующаяся величиной и размерностью. Ее использование `\hspace 1cm` некорректно, так как число 1, получаемое в качестве аргумента, размерности не имеет и потому *длиной* не является.

Описанный алгоритм действий компилятора называется *правилом наследования аргумента.*

Параметр служит для настройки каких-либо аспектов выполнения команды. При определении команды ему присваивается определенное значение. Если при выполнении команды оно не требует корректировки, параметр можно опустить вместе с квадратными скобками.

В качестве примера использования параметров и аргументов приведем команду:

`\rule[`*смещение*`]{`*длина*`}{`*толщина*`}` ,

рисующую линию заданной *длины* и *толщины* и смещенную при необходимости относительно базисной линии строки. Например, команда `\rule[5pt]{25pt}{1pt}` рисует смещенную вверх горизонтальную

линию ———, команда `\rule[-5pt]{1pt}{10pt}` — смещенную вниз вертикальную линию |, а команда `\rule{6pt}{6pt}` — несмещенный квадратик ■.

Таблица 1.1

Служебные символы

Команды	Назначение	Символы
`\`	признак команды	`\textbackslash`
`#`	параметр команды	`\#`
`{`	открывающая скобка	`\{`
`}`	закрывающая скобка	`\}`
`_`	нижний индекс	`\_`
`^`	верхний индекс	`\textasciicircum`
`$`	математическая мода	`\$`
`&`	разделение ячеек	`\&`
`%`	комментарий	`\%`
`~`	неразрывный пробел	`\textasciitilde`

Для часто выполняемых действий используются односимвольные команды, или служебные символы, упрощенный синтаксис которых является исключением из правил. Они собраны в первой колонке табл. 1.1, во второй описано их назначение, а в последней приведены команды, используемые, чтобы напечатать сами символы. Часть из них также имеют нестандартный синтаксис.

Отметим, что команды нижнего и верхнего индекса имеют аргумент, подчиняющийся приведенному выше правилу наследования. Сравните, `$x^2_ij$` ⇝ x^2_ij и `$x^2_{ij}$` ⇝ x^2_{ij}.

Большое число символов, используемых в тексте и формулах, заданы командами, например `\dag` ⇝ †. Потребность в них сокращается по мере расширения использования в LaTeX кодировки Unicode, но пока еще они являются весомым компонентом формул и порой необходимы в тексте. Следует учитывать, что компилятор автоматически уберет пробел, следующий за именем команды, поэтому символ, заданный командой и отделенный от последующего текста одним пробелом, сольется ним: `\ddag пример` ⇝ ‡пример. Чтобы избежать этого, команду-символ можно заключить в фигурные скобки — `{\ddag}...`, поставить пару скобок после нее — `\ddag{}...`, или задать пробел явно командой «`\ `» — `\ddag\ пример` ⇝ ‡ пример.

Одной из основных конструкций LaTeX является окружение:

```
\begin{имя окружения} аргументы и параметры
    область действия окружения
\end{имя окружения}
```

Оно формируется командами `\begin` и `\end`, охватывающими часть рукописи, являющуюся *областью действия окружения*. Их аргумент — имя окружения — определяет операцию, производимую над областью действия окружения. Для имен окружений справедливы требования, предъявляемые к именам команд.

Открывающая команда `\begin` может иметь дополнительные аргументы и параметры, а у закрывающей команды `\end` они отсутствуют, так как ее задача лишь ограничить область действия производимой операции. Окружения позволяют более наглядно представить структуру рукописи и в этом их основное назначение.

Некоторые команды имеют окружения-аналоги, например команда `{\itshape\`*курсив*`}` и окружение

```
\begin{itshape}
    Для верстки большого\\
    объема текста лучше\\
    пользоваться окружениями.
\end{itshape}
```

Обычно команды используются для обработки небольших фрагментов текста в пределах строки или абзаца, тогда как окружения более удобны при верстке нескольких абзацев.

Области действия команд тесно связаны с группами — одним из базовых понятий TeX. Прежде чем приступить к верстке, компилятор считывает множество определений команд и значений переменных, задающих их параметры. Лишь малая часть переменных является глобальными и еще меньшая — статическими, т. е. сохраняющими неизменные значения во время компиляции. Подавляющая же часть представляет собой локальные динамические переменные, значение которых определено только внутри группы.

Группой является аргумент и параметр команды, область действия окружения, математическая формула, ячейка таблицы или просто часть текста, заключенная в фигурные скобки `{...}` или между командами `\begingroup` и `\endgroup`. С точки зрения компилятора, сам текст рукописи также представляет собой группу, так как заключен в окружение document. Группы вкладываются друг в друга: рукопись содержит

окружения и команды, в аргументах и параметрах которых находятся другие окружения и команды со своими аргументами и параметрами, и т. д. Уровень вложенности может быть очень велик. При выходе из вложенной группы измененные локальные переменные восстанавливают свои значения, а вновь введенные переменные и команды становятся неопределенными.

Исходя из сказанного, команды, обрабатывающие текст, можно условно разделить на два типа. Область действия одних ограничена их собственным аргументом. Они создают группу (аргумент) и внутри нее меняют динамическую переменную, например параметр шрифта, как в этом случае: `\textbf{при}мер` ⇝ **при**мер. К данному типу команд относятся и окружения, область действия которых также является группой.

Команды другого типа, называемые *декларациями,* действуют в пределах группы, в которой находятся, начиная с позиции, в которой они стоят. Декларации не действуют на предшествующий текст, но изменяя значение какой-либо динамической переменной, меняют режим верстки следующего за ними текста, вплоть до выхода из группы.

Поясним сказанное примером, в котором группы формируются фигурными скобками, а декларации `\it`, `\sf`, `\large` и `\small` изменяют динамические переменные, управляющие начертанием и размером шрифта. Совместив код с результатом его выполнения, выделим группирующие скобки и команды мелким шрифтом и подчеркиванием.

`{` Для начала создадим группу, вложенную в обычный текст. `\it\large` *Здесь изменим шрифт на курсив большого размера.* `{\sf\small` Создадим вложенную группу и установим рубленный шрифт малого размера. `}` *Выйдем из вложенной группы. Восстановился большой курсив.* `}` А теперь, выйдя из первой группы, возвратимся к исходному шрифту.

Некоторые декларации изменяют значение глобальных переменных, область действия которых не ограничивается группой. Вносимые ими изменения действуют до конца компиляции, или до новой установки значений переменных с помощью этих деклараций. Такие декларации будем называть *глобальными.*

1.5. Структура документа

Как правило, документы LaTeX состоят из титульной части, разделов, рисунков, таблиц и списков литературы. В книгах к ним добав-

ляются оглавления и предметные указатели. Создаваемая структура связана нумерацией объектов типа формул, разделов, цитируемых источников и т. п., и ссылками на них в тексте документа.

1.5.1. Преамбула. Класс документа и пакеты

Текст рукописи начинается с *преамбулы*, в которой задается класс документа, перечисляются загружаемые пакеты, настраиваются существующие команды и определяются новые. Завершает преамбулу команда `\begin{document}`.

Команда

`\documentclass[`*параметры*`]{`*класс документа*`}[`*версия*`]`

загружает класс документа и осуществляет настройку основных параметров верстки. Класс можно задать только один, а параметры перечисляются списком через запятую. Например, для статьи, набираемой в две колонки шрифтом 12pt, укажите

`\documentclass[a4paper,12pt,twocolumn]{article}`.

В последнем параметре можно запросить *версию* класса в форме «год.месяц.число», например, 2023.03.20. Если версия загружаемого класса окажется более ранней, будет выдано предупреждение.

Научные издательства разработали большое число собственных классов, соответствующих их стилю и полиграфическим традициям. В основном они базируются на стандартных классах LaTeX [11], конструкции которых разработчики адаптируют под свои нужды, добавляя при необходимости небольшое число новых. Синтаксис основных конструкций и команд LaTeX, как правило, не меняется при замене одного класса документа на другой, поэтому изучение классов и программных средств стандартного LaTeX позволяет в дальнейшем самостоятельно освоить и использовать любой новый класс документа. Перечислим основные стандартные классы и параметры их настройки.[3]

Стандартные классы LaTeX

`article` — наиболее часто используемый класс, предназначенный для верстки научных статей.

`report` — класс, предназначенный для верстки отчетов.

[3] Для верстки писем и презентаций имеются классы `letter` и `slides`, практически не используемые в настоящее время, поэтому мы опустили их.

proc — класс, предназначенный для верстки трудов конференций. Основан на классе standard, но отличается тем, что текст печатается в две колонки.

book — класс, предназначенный для верстки книг. Характеризуется наиболее полным списком разделов. Обычно имеет оглавление, указатель, колонтитулы и двусторонний вывод с разными ширинами полей четных и нечетных страниц.

Параметры настройки стандартных классов

10pt | 11pt | 12pt — основной размер шрифта документа.

letterpaper | legalpaper | exclusivepaper | **a4paper** | a5paper | b5paper — размер страницы. По умолчанию используется параметр, заданный в конфигурации дистрибутива TEX, a4paper в Европе и letterpaper в Америке. Параметры a5paper и b5paper не поддерживаются классом proc.

landscape — альбомная ориентация страницы. По умолчанию используется портретная ориентация.

draft | **final** — маркировка неполных или переполненных строк. В режиме final проблемные строки не маркируются.

oneside | twoside — размеры полей четных и нечетных страниц. По умолчанию для всех классов, кроме book, используется oneside — поля одинаковы. Параметр twoside не используется в классе slides.

openright | openany — стиль разбиения на главы, которые могут начинаться только с правой (openright) или с любой (openany) страницы. По умолчанию первый метод использует класс book, а второй — report.

onecolumn | twocolumn — вывод текста в одну и две колонки. По умолчанию все классы, кроме proc, верстают в одну колонку.

titlepage | notitlepage — вывод заголовка и аннотации. По умолчанию в классах proc и book заголовок и аннотация выносятся на отдельную страницу, что соответствует titlepage, а в классе article они располагаются в начале документа. Параметр titlepage не поддерживается классом proc.

fleqn — выравнивание формул по левому краю. По умолчанию они центрируются.

leqno | **reqno** — по умолчанию номера печатаются справа от формул, а при использовании параметра leqno они располагаются слева.

openbib — верстка списка литературы в «открытом стиле», в котором логические блоки цитируемого источника, разделяемые командами `\newblock`, начинаются с новой строки.

Возможности LaTeX существенно расширяются библиотеками-пакетами, которые выполняют дополнительные настройки команд и конструкций, позволяют оперировать графическими объектами и гипертекстовыми ссылками, вводят команды для реализации новых действий и т. п. В настоящее время насчитывается более тысячи пакетов и число их неуклонно растет, так как каждый пользователь вправе написать свой пакет, сделать к нему описание, зарегистрировать и выложить в CTAN-архив.[4]. Описание многих пакетов содержат книги [2–6], а свежую документацию ко всем пакетам можно загрузить с помощью MikTex (см. разд. А.3)

Пакеты нужно разделить на стандартные и нестандартные. Первые реализуют действия, необходимые для всех классов документов, они рекомендованы к общему употреблению, так как не конфликтуют между собой и со стандартными классами. Нестандартными пакетами, многие из которых уже не поддерживаются, можно пользоваться только на свой страх и риск, и хотя некоторые из них упрощают верстку, их не стоит включать в свой арсенал, так как они являются потенциальными источниками ошибок компиляции.

Загружает и настраивает пакет команда

`\usepackage[`*параметры*`]{`*имя пакета*`}[`*версия*`]`

Она аналогична команде `\documentclass` с той лишь разницей, что в ее аргументе можно указать список пакетов, перечисленных через запятую, но загружать списком следует лишь пакеты, не имеющие настроек.

Пакет загружается только один раз, при повторной попытке компилятор выдает предупреждение, что это уже сделано. Сообщения о загрузке пакетов и предупреждения компилятора выдаются на консоль и записываются в log-файл.

Классы и пакеты могут самостоятельно (неявно) загружать и настраивать другие необходимые для их работы пакеты. Если неявно загружаемый пакет нужно настроить по иному, сделайте это, загрузив его командой `\usepackage`, *до того*, как это делает вызывающий его

[4] CTAN (Comprehensive TeX Archive Network) — сеть общедоступной информации по TeX'у можно найти в интернете по адресу: `www.ctan.org`

пакет. При этом не забудьте добавить к своим настройкам и те, что устанавливает вызывающий пакет, иначе часть его действий может быть нарушена. Списки загружаемых пакетов и их настроек приводятся в документации вызывающих пакетов.

Классы документов и некоторые пакеты передают параметры, указанные при их загрузке и неиспользованные для собственной настройки, загружаемым после них пакетам. Это также можно использовать для настройки неявно загружаемых пакетов.

Приведем пример простейшего документа.

```
\documentclass{article}
\usepackage[utf8]{inputenc}
\usepackage[russian]{babel}
\begin{document}
  Текст документа.
\end{document}
Место для заметок и вариантов.
```

Текст документа.

Верстаемый текст располагается после преамбулы в окружении document. Обратите внимание, что фраза «Место для заметок и вариантов.» отсутствует в сверстанном документе. Текст, находящийся за командой `\end{document}`, не обрабатывается. В следующем за ней «остатке» файла можно хранить заготовки или варианты текста рукописи, чтобы использовать их в дальнейшем. Если компиляция дает большое число ошибок, с которыми трудно разобраться, ограничьте компилируемую часть рукописи, переместив команду `\end{document}` ближе к ее началу. Это облегчит работу над ошибками.

```
\documentclass[12pt]{article}                параметры верстки,
\usepackage[a4paper,%                          размер страницы,
  text={180mm, 260mm},%                  ширина и высота текста,
  left=15mm, top=15mm]{geometry}       размеры пустых полей,
\usepackage[utf8]{inputenc}                  кодировка текста,
\usepackage{cmap}                          кодировка pdf-документа,
\usepackage[russian]{babel}              поддержка русского языка,
\usepackage{indentfirst}                  корректировка отступов,
\usepackage{amssymb}                    математические символы,
\usepackage{amsmath}                  математические конструкции,
\usepackage{graphicx}                        поддержка графики.
```

Приведенная выше минимальная преамбула, обеспечивает поддержку русского языка, графики и расширенные ресурсы верстки формул. Справа дан краткий комментарий о назначении пакетов, а более по-

дробно они обсуждаются по мере необходимости. Рассматриваемые далее возможности верстки предполагают загрузку всех этих пакетов.

1.5.2. Параметры страницы

Удобный и простой способ управления параметрами страницы дает пакет geometry. Число его настроек очень велико [12]. Перечислим лишь те, что используются наиболее часто.

landscape | **portrait** — ориентация страниц.

a4paper | a5paper | b5paper | letterpaper ... — стандартный размер страниц. Определены практически все известные стандарты.

paperwidth =..., paperheight =...| papersize ={ширина, высота} — страницы произвольного размера. При использовании papersize в фигурных скобках указывается ширина и высота страниц, например `papersize={100mm,150mm}`.

textwidth =..., textheight =...| text ={ширина, высота} — ширина и высота текста.

left =..., right =...| hmargin ={слева, справа} | hcentering — ширина левого и правого поля. При использовании hcentering поля имеют одинаковую ширину.

top =..., bottom =...| vmargin ={сверху, снизу} | vcentering — высота верхнего и нижнего поля. При использовании vcentering поля имеют одинаковую высоту.

centering — эквивалент одновременного использования hcentering и vcentering.

total ={ширина, высота} — полная ширина и высота всех компонент страницы.

includehead | includefoot | includeheadfoot — включение верхнего или нижнего, или обоих колонтитулов в высоту total.

onecolumn | twocolumn — вывод текста в одну и две колонки.

columnsep =... — ширина поля, разделяющего колонки текста.

footnotesep =... — высота зазора между нижнем краем текста и сносками.

marginparwidth =...| marginpar =... — ширина поля заметок на полях.

`marginparsep` =... — ширина поля, отделяющего заметки на полях от текста.

`nomarginpar` — отсутствие заметок на полях.

О колонтитулах и заметках на полях см. разд. 8.8.

1.5.3. Титульная часть

В начале рукописи обычно располагаются команды, содержащие заголовок, список авторов, дату создания и аннотацию документа. За ними следует команда `\maketitle`, обеспечивающая вывод этих сведений в верстаемый документ:

```
\begin{document}
\title{Заголовок}
\author{Автор, или перечень авторов}
\date{Дата создания документа}
\begin{abstract}
     Аннотация ...
\end{abstract}
\maketitle
```

Открывает документ заголовок, помещаемый в аргумент команды

`\title{`*заголовок*`}` .

Длинный заголовок можно поделить на строки командами `\\`.

Список авторов вместе с перечислением названий организаций и их адресов указывается в аргументе команды

`\author{`*список авторов*`}` .

Длинный список также можно поделить на строки командами `\\`, или, разбив его на группы, использовать несколько команд `\author`.

В большинстве классов после списка авторов автоматически выводится текущая дата, а если это не предусмотрено, ее можно напечатать, воспользовавшись комбинацией команд `\date \today` , первая печатает дату, генерируемую второй командой. Команда `\date{`*дата*`}` позволяет ввести дату вручную или подавить автоматический вывод, указав пустой аргумент.

Компилятор игнорирует команды `\title`, `\author` и `\date`, если в рукописи отсутствует команда `\maketitle` , формирующая титульную часть документа. Оформление титульной части отличается от остального документа, например при двухколоночном наборе она занимает ширину страницы, а не колонки, а при использовании параметра

titlepage стандартного класса выносится на отдельную страницу. Команда `\maketitle` указывает позицию, отделяющую титульную часть от остального текста. Она должна стоять после команд `\title`, `\author` и `\date`.

Окружение abstract формирует аннотацию. Ее положение зависит от класса документа. Во многих классах (например, revtex4-2 [13, 14] Американского физического общества и elsarticle издательской корпорации Elsevier [15]) она входит в титульную часть и потому должна находиться перед `\maketitle`. Стандартный класс article создаст титульную страницу с аннотацией, если она находится перед `\maketitle`; и поместит ее после титульной части в начале первой колонки текста, если окружение abstract стоит после команды `\maketitle`.

1.5.4. Разделы

Документ LaTeX разбивается на ряд смысловых частей, которые формируются в виде разделов документа. Разделы разбиваются на семь уровней, среди которых есть части, главы, разделы, параграфы и и т. д.. Число доступных уровней зависит от класса документа. Далее для всех уровней мы будем применять обобщающее наименование «разделы».

Таблица 1.2

Команды разделов

Команды	Уровни	book	report	article	proc	Команды
`\part[•]{•}`	−1	∘	∘			`\part*{•}`
`\chapter[•]{•}`	0	∘	∘			`\chapter*{•}`
`\section[•]{•}`	1	∘	∘	∘	∘	`\section*{•}`
`\subsection[•]{•}`	2	∘	∘	∘	∘	`\subsection*{•}`
`\subsubsection[•]{•}`	3	∘	∘	∘	∘	`\subsubsection*{•}`
`\paragraph[•]{•}`	4	∘	∘	∘	∘	`\paragraph*{•}`
`\subparagraph[•]{•}`	5	∘	∘	∘	∘	`\subparagraph*{•}`

Уровни разделов стандартных классов и создающие их команды представлены в табл. 1.2. Ее первая и последняя колонки содержат команды, а кружочки отмечают возможность их использования в документах данного класса. Во второй колонке указаны числовые идентификаторы уровней. Как видно из таблицы, наиболее широким на-

бором разделов обладают классы book и report. В книгах основным уровнем являются главы, которые могут разбиваться на разделы уровня \section и более мелкие части, или объединяться в части \part. Статья разбивается на разделы уровня \section и ниже.

Команды разделов перечислены в таблице в порядке вложенности уровней. Разделы верхних уровней могут содержать любое количество разделов более низких уровней, при этом с каждым уровнем ассоциирован счетчик. При создании раздела к счетчику его уровня добавляется единица, а счетчики более низких уровней сбрасываются, и нумерация всех вложенных разделов начинается заново. Таким образом, LaTeX автоматически нумерует главы, разделы, параграфы и т. д. Если непосредственно после команды раздела поставить команду \label{*имя метки*} она присвоит метке с данным *именем* значение номера раздела.

Чтобы создать раздел, достаточно поставить в его начале соответствующую команду и пометить ее:

```
\section{Слоны - мои друзья!}\label{s:слоны}.
```

Первая команда генерирует номер и выводит его вместе с заголовком, а вторая обеспечивает возможность ссылки на него.

Команды разделов имеют аргумент и параметр, например:

```
\chapter[заголовок в оглавлении]{заголовок в тексте}.
```

В аргумент помещается *заголовок*, печатаемый в тексте, а в параметр — используемый в оглавлении и колонтитулах. Например, команда

```
\section[О друзьях]{Слоны - мои друзья!}\label{s:слоны},
```

напечатает «Слоны — мои друзья!» и отправит «О друзьях» в оглавление. В аргументе длинный заголовок можно разбить на строки командами \\, но в параметре этого делать не следует. При отсутствии параметра в оглавление и колонтитулы заносится заголовок аргумента.

Команды разделов имеют дополнительный вариант с именем, завершающимся символом *. Команды «со звездочкой» предназначены лишь для печати заголовков. Они не имеют параметра и не создают новые разделы. Номера перед заголовками не ставятся, значения счетчиков не меняются, а напечатанные заголовки не попадают ни в оглавление, ни в колонтитулы.

Декларация \appendix отделяет приложения от основной части документа. Ее действие демонстрирует приведенный ниже пример. Она сбрасывает все счетчики, начиная с уровня \chapter в классах book и report, или \section в классах article и proc. Кроме этого несколько

изменяется действие самих команд \chapter или \section. Во-первых, для вывода значений их счетчиков используются заглавные латинские буквы. Во-вторых, при создании приложения сначала выводится заголовок со стандартным названием «Приложение» (Appendix) и номером, а потом — название раздела без номера.

```
\appendix
\section{Пример}\label{•}
    Текст приложения...
\subsection{Заголовок}\label{•}
    Продолжение...
```

Приложение A

Пример

Текст приложения...

A.1 Заголовок

Продолжение...

1.5.5. Оглавление

Команда \tableofcontents инициирует создание оглавления. При ее наличии компилятор, собрав названия разделов вместе с номерами страниц, на которых они созданы, формирует оглавление и сохраняет его в файле с расширением •.toc. При последующих компиляциях команда \tableofcontents замещается заголовком «Оглавление» (Table of Contents) и считанным из toc-файла оглавлением, а сам файл обновляется. Таким образом, изменение структуры разделов рукописи отразится в оглавлении документа не сразу, а лишь на втором проходе компиляции.

1.5.6. Метки, ссылки, счетчики

LaTeX имеет универсальную команду-метку:

\label{*имя метки*},

которой можно пометить объект, чтобы ссылаться на него. Ее аргументом является *имя метки*, а объектом может быть формула, раздел, рисунок, таблица и просто позиция в тексте, в которой находится эта команда. Метке автоматически присваивается текущее значение счетчика страниц и, в зависимости от контекста, счетчиков формул, рисунков и и т. д. Информация о значениях меток заносится в •.aux-файл. Сослаться на метку позволяют команды:

\ref{*имя метки*} и \pageref{*имя метки*}.

Первая из них выводит номер помеченного объекта, а вторая — номер страницы, на которой он находится. Имеется также команда

`\eqref{`*имя метки*`}`,

предназначенная для вывода номера формулы, который она автоматически заключает в круглые скобки. Загрузка пакета hyperref преобразует ссылки в гиперссылки (см. разд. 7.8).

Пакет varioref коллекции tools расширяет возможности ссылок. Он вводит команду `\vref{`*имя метки*`}`, одновременно печатающую и номер объекта, и номер страницы, при этом номер страницы не выводится, если ссылка и объект находятся на одной странице. Например, `|fig.~\vref{f:SomeFigure}|` ⇝ fig. 3 on page 12. Аналогично действует команда `\vpageref{`*имя метки*`}`, печатающая только номер страницы. Перед номерами страниц автоматически выводятся префиксы «on page», «on the following page», «on the next page», «on the preceding page» и т. д. Префиксы, составленные для многих языков, подключаются с помощью параметров пакета varioref [16]. В частности, для настройки русских ссылок его нужно загрузить с параметром russian.

Ранее упоминалось, что для связывания меток и ссылок используются данные, хранящиеся в •.aux-файлах. Поэтому для «разрешения» ссылок необходимы как минимум две компиляции: при первой список значений меток сохраняется в файл, а при второй команды ссылок извлекают эту информацию. Если метка размещена в «плавающем» объекте (см. разд. 3), для корректного разрешения ссылок требуется три прохода компиляции: на первом метка получает значение счетчика, соответствующее порядку следования объекта в рукописи; на втором оно корректируется с учетом его положения в сверстанном документе; и лишь на третьем номер, выводимый в документ, становится правильным.

Рисунки, таблицы, формулы и другие объекты автоматически подсчитываются во время компиляции. Каждый объект имеет собственный счетчик, обычно называемый так же, как и сам объект.

Таблица 1.3

Список основных счетчиков

Разные		Разделы		Списки
page	footnote	part	subsubsection	enumi
figure	mpfootnote	chapter	paragraph	enumii
table	parentequation	section	subparagraph	enumiii
equation		subsection		enumiv

Список основных счетчиков приведен в табл. 1.3. В первой колонке перечислены счетчики страниц, рисунков, таблиц и уравнений, во

второй — счетчики сносок в основном тексте, footnote, и на министраницах, mpfootnote (см. разд. 2.5), а также счетчик связанных уравнений parentequation, имеющих общий номер equation. В третьей и четвертой колонках представлены счетчики разделов, а в пятой колонке находятся счетчики нумерованных списков с первого до четвертого уровня (см. разд. 5.1 и 8.10.2).

Присвоить счетчику значение или сдвинуть его на нужную величину позволяют команды:

```
\setcounter{счетчик}{значение}.
\addtocounter{счетчик}{приращение},
```

К примеру, так можно сбросить счетчик примечаний:

```
\setcounter{footnote}{0}.
```

Отрицательное *приращение* уменьшает значение счетчика, поэтому команда `\addtocounter{page}{-10}` вернет нумерацию на десять страниц назад.

Текущее значение счетчика выводит команда `\the`*счетчик*. Для независимых счетчиков выводятся только их значения, например `\thepage` ⇝ 33. К значениям подчиненных счетчиков обычно впереди добавляются значения внешних. Например, номер данного раздела `\thesubsection` ⇝ 1.5.6, а значения внешних счетчиков равны `\thesection` ⇝ 1.5 и `\thechapter` ⇝ 1. Подробнее о подчиненности счетчиков см. разд. 7.4.

Некоторые команды, связанные с обработкой счетчиков и ссылок на них, являются *хрупкими.* В тексте они обрабатываются корректно, но, находясь в аргументах или параметрах других команд и окружений,[5] могут вызвать ошибки компиляции.

Ситуация с некорректной обработкой хрупких команд может возникать, если они оказываются в «перемещаемом» аргументе, который записывается в один из служебных файлов и используется при последующих компиляциях (см. рис. 1.3). Коллизии с хрупкими командами можно ожидать в названиях разделов, в подписях рисунков и таблиц, в аргументах команд, находящихся в плавающих объектах.

По мере совершенствования кода LaTeX список хрупких команд сокращается, и потому мы его не приводим, но некоторые из них обсуждаются далее. Команда `\protect`, изменяя алгоритм действий компилятора, позволяет «защитить» хрупкие команды и устранить ошибки. Она не имеет аргументов и ставится непосредственно перед

[5] В случае окружений это справедливо также и для области их действия.

«защищаемой» командой. Ее можно использовать и с обычными командами, кроме тех, что хранят длины, или изменяют значения счетчиков. Это не принесет вред, однако может избавить от ошибок компиляции, если они не связаны с несовместимостью команд, порой возникающей при работе с нестандартными пакетами.

1.5.7. Примечания

Примечания к тексту оформляются в виде сносок, выводимых внизу страницы.[6] Текст сноски помещается в аргумент команды

`\footnote[`*номер*`]{`*текст*`}`.

Сноски подсчитываются и нумеруются. По умолчанию новой сноске присваивается текущее значение счетчика footnote, но параметр команды `\footnote` позволяет указать любой другой номер или символ. В стандартных классах book и report сноски нумеруются по главам, а в остальных используется сквозная нумерация. Примечания министраниц (см. разд. 2.5) нумеруются отдельно с помощью собственного счетчика mpfootnote.

Команда `\footnoote` генерирует в тексте ссылку и печатает сноску внизу страницы, однако не всегда получается выполнить обе эти операции вместе. При создании некоторых объектов связь с текущей страницей отсутствует, поэтому сноску вывести некуда. Например, в таблице tabular или плавающем объекте (см. гл. 3) команда `\footnoote` генерирует лишь ссылку, а сама сноска теряется. В таких случаях примечание формируется в два этапа с помощью команд:

`\footnotemark[`*номер*`]`,
`\footnotetext[`*номер*`]{`*текст*`}`.

Команда `\footnotemark` используется внутри объекта, чтобы сгенерировать ссылку, а команда `\footnotetext` ставится снаружи непосредственно после объекта, чтобы вывести текст сноски.

Команда `\footnotemark` является хрупкой, и если она вызовет проблемы, защитите ее командой `\protect`.

[6] Сноски отделяются от основного текста чертой и печатаются мелким шрифтом.

Глава 2

Верстка текста

Верстая документ, компилятор решает задачу оптимального распределения объектов на площади страницы. Объекты представляют собой прямоугольники, в терминологии TEX называемые *боксами.* Боксы привязываются к базисным горизонтальным линиям и разделяются пробелами. Таким образом формируются строки.

Боксы могут иметь различную внутреннюю структуру. Простейший из них представляет собой букву или слово. Боксами являются также и загружаемая из файла иллюстрация  , таблица

a	c
e	n

или строчная формула $\int x\,dx = x^2/2 + C$. Структура бокса может быть очень сложна, но будучи сформирован, он встраивается в текстовую строку как простой прямоугольник. Это означает, что обсуждаемые далее приемы верстки применимы не только непосредственно к тексту, но и для управления положением иллюстраций, таблиц и т. п.

Текст разбивается на абзацы. Верстая абзац, компилятор нарезает строки одинаковой длины. Заполняя строку, он «нанизывает» боксы на базисную линию до тех пор, пока их длина вместе с пробелами не превысит длину строки. Если крайний бокс можно разделить, часть его будет перенесена на следующую строку. Если же он представляет собой единое целое, компилятор оценит возможность опустить его в следующую строку, увеличив пробелы в текущей.

Сверстанный абзац встраивается в текущую страницу и, если она переполняется, часть строк переносится на новую страницу. Сначала на ней окажется мало строк, поэтому компилятор попытается перенести

часть текста с предыдущей страницы, увеличив интервал между абзацами, оценит возможность перераспределить текст за счет изменения позиций плавающих объектов (см. разд. 3.3 и 8.6) и т. д.

В процессе компиляции оптимизируется не только результат выполнения отдельных операций, но и верстки в целом. Добавление одной строки может изменить распределение текста на нескольких страницах, поэтому пока документ не набран полностью, не стоит обращать внимание на погрешности автоматической верстки. Исправлять их следует в самом конце, окончив редактирование самого текста. Именно эта работа является главной при подготовке публикации.

2.1. Длины

Боксы, строки, страницы и другие объекты имеют размеры, измеряемые в «длинах». Между буквами, словами, строками, абзацами и т. д.имеются пустые промежутки, или пробелы, каждый из которых также имеет какую-то длину. Длина LaTeX является очень гибкой величиной. Длины могут быть абсолютными и относительными, фиксированными и упругими.

Простейшей является фиксированная длина, задаваемая значением и размерностью. Основные размерности длин приведены в табл. 2.1. Значение представляет собой действительное число, в котором можно опустить нулевую целую или дробную часть, например `1cm`, `+1.5pt` или `-.2mm`. При этом единичное значение обязательно указывается, так как сама размерность длиной не является.

Упругие длины имеют *естественную длину, сжимаемость и растяжимость,* указываемые следующим образом: `12pt plus 4pt minus 4pt`. Такая длина может изменяться от `8pt` до `16pt`. При верстке сначала будет опробована ее естественная длина, а при необходимости она будет увеличена или уменьшена в указанных пределах. Использование упругих длин в пробелах, разделяющих боксы и абзацы, позволяет оптимизировать заполненность строк и страниц.

Относительные длины задаются в единицах `em` и `ex`, привязанных к текущему размеру шрифта, или кеглю, как принято называть его в полиграфии. Величина `1em` соответствует ширине заглавной буквы «M», а `1ex` — высоте прописной буквы «x». Абсолютная величина относительной длины вычисляется в соответствии с текущим кеглем при ее использовании. В конструкциях LaTeX относительные длины позволяют согласованно регулировать размер шрифта, пробелов и

межстрочного интервала, который задается длиной, меняющейся с изменением кегля.

Таблица 2.1

Основные единицы длин

Единицы	Значения	Единицы	Значения
mm	миллиметр (2.845 pt)	pt	пункт (0.351 mm)
cm	сантиметр (0.39598 in)	in	дюйм (25.40 mm)
em	ширина буквы «M» текущего шрифта	ex	высота буквы «x» текущего шрифта

Таблица 2.2

Основные длины

Параметры страницы		Абзацы	Разные
`\paperwidth`	`\textwidth`	`\parindent`	`\baselineskip`
`\paperheight`	`\textheight`	`\parskip`	`\columnsep`
`\voffset`	`\hoffset`	`\linewidth`	`\fill`

Чаще всего длины хранятся в командах. Во время компиляции определен ряд длин разного масштаба. Те из них, что используются наиболее часто, приведены в табл. 2.2. Первые две колонки содержат ширину и высоту страницы (`\paperwidth` и `\paperheight`) и текста (`\textwidth` и `\textheight`), а также длины, позволяющие регулировать высоту верхнего (`\voffset`) и ширину левого (`\hoffset`) пустого поля на краю станицы.

В третьей колонке находятся ширина отступа в начале абзаца `\parindent` , величина дополнительного интервала между абзацами `\parskip` и межстрочный интервал `\baselineskip` . В последнюю колонку собраны ширина строки `\linewidth` , которая при верстке в две колонки отличается от ширины текста, ширина зазора между колонками `\columnsep` и длина `\fill` , обладающая нулевой естественной величиной и бесконечной растяжимостью. Примеры использования длин приводятся далее.

Изменить длины, хранящиеся в командах, позволяют декларации:

```
\setlength{команда}{значение} ,
\addtolength{команда}{добавка} .
```

Первая устанавливает новое *значение* длины, а вторая изменяет ее на величину *добавки*. Команда `\the`*длина* печатает значения *длин*, например `\the\linewidth` ↝ 321.51613pt.

2.2. Верстка абзацев

Рассмотрим алгоритм верстки абзацев и некоторые методы изменения формата строк. Прежде всего отметим, что абзацы могут разделять не только пустые строки, но и команды `\par`.

Уже говорилось, что строки формируются из боксов, разделяемых упругими пробелами, как показано на рис. 2.1. На рисунке сплошными линиями обозначены границы боксов, а также базисная линия и границы строки. Вверху показаны параметры бокса с привязкой к базисной линии и место разбиения последнего бокса в соответствии с таблицей переноса.

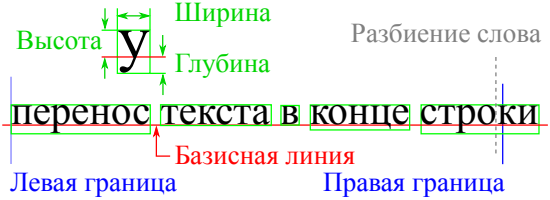


Рис. 2.1. Структура строки текста

Каждый бокс имеет высоту, ширину и глубину. При встраивании в строку бокс опускается относительно базисной линии на свою глубину, а его высота задает размер над базисной линией.

Простейший бокс — символ, например буква. Набранное слово также становится боксом, высота и глубина которого соответствуют максимальной высоте и глубине составляющих его букв. Буквы «склеиваются» упругим «клеем», поэтому длина бокса-слова может отличаться от суммарной ширины букв. Некоторые буквы автоматически сближаются, чтобы промежуток между ними не выглядел слишком рыхлым. Изменение интервала между буквами в зависимости от их формы называется *кернингом*. Сравните слово «улица» с его вариантом без кернинга `\mbox{у}лица` ↝ улица. В авторской терминологии TEX [1] все пустые промежутки называются *клеем*, но мы для определенности будем разделять их на пробелы, имеющие ненулевую естественную длину, и пружины, естественная длина которых равна нулю, но растяжимость и сжимаемость отличны от него.

В абзаце, состоящем из текста, набранного шрифтом одного размера, межстрочный интервал одинаков. Однако расстояние между строками, содержащими боксы с большой высотой и глубиной, увеличивается в соответствии с размерами боксов, как это показано на с. 35. Алгоритм верстки абзацев не допускает, чтобы межстрочный интервал оказался

меньше, чем глубина предыдущей и высота последующей строк, если сам автор не изменяет расстояние между ними.

2.2.1. Выравнивание строк

Верстая абзац, TEX выравнивает его левую и правую границу. Нарезая текст на строки одинаковой длины, он использует таблицу переносов и автоматически разбивает слова, оказавшиеся в конце строки.

Таблицы переносов предоставляет пакет `babel`. Язык указывается в параметре загружающей его команды `\usepackage`. Таблица переносов используется для разбиения слов. Если перенос выбран неудачно, или слово отсутствует в таблице, перенос нужно настроить вручную, указав командами `\-`, как можно разбить слово (например, `абра\-ка\-даб\-ра`). Если слово окажется в конце строки, будет выбран оптимальный перенос, в остальных же случаях команды `\-` игнорируются. Продемонстрируем это следующим образом. Длинное «слово» *xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx*, заведомо отсутствующее в таблице переносов, разобьем на группы из трех букв и получим вполне приемлемый результат: `xxx\-xxx...\-xxx\-xxx` ⇝ *xxxxxxxxxxxxx-xxxxxxxxxxxxxxxxxxxxx*.

Иногда, напротив, требуется подавить перенос, или сделать небольшой фрагмент текста «монолитным». Для этого его нужно поместить в бокс командой

`\mbox{`*текст*`}`.

В таком боксе можно разместить одно или несколько слов, формул и других объектов. Если длина бокса окажется велика, компилятор попробует перенести его в следующую строку, увеличив пробелы в текущей, но если она окажется очень «рыхлой», TEX, сжав пробелы до минимума, оставит бокс в текущей строке, которая из-за этого может вылезти за границу страницы.

В ряде случаев, чтобы слова не оказались в разных строках, их следует «склеить» неразрывным пробелом `~`. Это нужно делать в сокращениях «`т.~е.`», «`г.~Петров`», размерных величинах «`10~В/см`», а также в ссылках на номера страниц «`с.~\pageref{p:SomePage}`» и объектов «`на рис.~\ref{f:SomeFigure}`».

Встретив в конце строки дефис (-), короткое (–) или длинное (—) тире, компилятор сочтет их оптимальными точками разбиения строки. Предотвратить разбиение в таком случае поможет команда `\nobreakdash`. Формально она предназначена для подавления перено-

са в конструкциях типа диапазона значений, `7\nobreakdash--10` ⇝ 7–10, и подобных им, но может использоваться и в других случаях, когда требуется «склеить» объекты, не вставляя пробел.

Рассмотрим теперь, как можно принудительно разрывать строки. Команды перевода строки

`\\[длина]` и `\newline`

делает это, не заботясь о ее выравнивании. В параметре команды `\\` можно задать *длину*, которая добавляется к межстрочному интервалу. При ее положительном значении следующая строка отодвигается, а при отрицательном — приближается к разрываемой.

Разорвать строку, выровняв ее, позволяет команда

`\linebreak[число]` .

При принудительном выравнивании строка может оказаться очень «рыхлой», поэтому параметр команды позволяет настроить это действие. Он задается целым числом, изменяющимся в пределах от 0 до 4, регулирующим «жесткость» указания компилятору от рекомендации (0), до обязательного выполнения (4). По умолчанию, используется значение 4, т. е. строка разрывается и выравнивается при любой заполненности.

В стандартных классах, загружаемых с параметром draft, строки с плохой заполненностью помечаются небольшим прямоугольником.

2.2.2. Отступы

В начале абзаца, как правило, автоматически ставится небольшой пробел, называемый *отступом*. В стандартных классах первый абзац раздела делается без отступа, а в остальных абзацах он есть. В традициях русской полиграфии все абзацы начинаются отступом. Такую расстановку обеспечивает пакет indentfirst [17]. Его нужно загружать в каждом документе, написанном по-русски. Регулировать наличие отступа позволяют команды `\indent` и `\noindent` , которые ставятся в самом начале абзаца. Первая из них вставляет, а вторая, наоборот, убирает отступ. Длину отступа хранит команда `\parindent` .

2.2.3. Пробелы

Компилятор устанавливает величины пробелов в зависимости от заполненности строки, добиваясь оптимальной плотности текста. Если

же возникает необходимость вмешаться в его действия и вставить пробел нужной длины, это можно сделать несколькими способами.

Часто используемые пробелы определены в виде команд, приведенных в табл. 2.3. Их длины установлены в относительных единицах, поэтому при изменении размера шрифта они изменяются согласованно с окружающим текстом. Горизонтальные линии демонстрируют величины пробелов, соответствующих шрифту данного текста.

Таблица 2.3

Часто используемые пробелы

Команды	Длины	Команды	Длины
`\qquad`	2 em	`\quad`	1 em
`\enspace`	1/2 em	`\enskip`	1/2 em
`\;`, `\thickspace`	1/4 em	`\negthickspace`	−1/4 em
`\:`, `\medspace`	1/5 em	`\negmedspace`	−1/5 em
`\,`, `\thinspace`	1/6 em	`\!`, `\negthinspace`	−1/6 em
`\ `, `\space`	3.33 pt plus 1.67 pt minus 1.11 pt		

В последней строке приведен пробел `\ ` или `\space`, задаваемый упругой длиной, минимальная величина которой соответствует горизонтальной линии, а точкой показано ее максимальное удлинение. Напомним, что пробел, отделяющий команду от последующего текста, «съедается» при компиляции. Команды «`\ `» и `\space` помогают восстановить его, например «`1\AA равен` ⇝ 1Åравен» и «`1\AA\ равен` ⇝ ⇝ 1Å равен». Ширину такого пробела компилятор регулирует сам.

Отметим, что тонкий пробел `\,` является неразрывным. Он «склеивает» два объекта подобно символу `~`.

Вставить пробел любой длины позволяют команды

`\hspace{`*длина*`}` и `\hspace*{`*длина*`}`.

Если длина пробела положительна, следующий за ним текст сдвигается вправо, а при отрицательной длине — влево. С помощью отрицательного пробела можно наложить одну часть текста на другую, что не запрещено. Для примера, используя такое наложение, сделаем новый символ: `=\hspace*{-.63em}/` ⇝ ≠.

TEX отбрасывает пробелы, находящиеся в начале и конце строки, в том числе заданные командой `\hspace` и командами из табл. 2.3, а вот команда `\hspace*` выполняется всегда. С ее помощью можно даже вылезти за левое поле, как показано в следующем примере:

↶ `\\[-1ex] \hspace*{-2em}\tiger[2em].`

Этот трюк выполнил набор из трех команд, показанных справа. Первая перевела строку, сместив следующую вверх, чтобы скомпенсировать увеличение межстрочного интервала из-за большой высоты картинки, вторая сдвинула начало строки на -2em, а третья загрузила картинку шириной 2em.[1]

2.2.4. Пружины

Вдобавок к пробелам имеются «пружины» `\hfil` и `\hfill`, способные заполнить все свободное пространство строки. Пружины являются пробелами, обладающими нулевой естественной длиной и бесконечной растяжимостью. LaTeX неявно использует их для выравнивания текста. Например, при разрыве строки командой `\\` в ее конце ставится пружина `\hfil`, прижимающая текст влево. В стандартных конструкциях LaTeX, как правило, используются пружины `\hfil`, а команды `\hfill` позволяют пользователю корректировать выравнивание по своему усмотрению. При совместном использовании пружина `\hfill`, имеющая большую жесткость, «сминает» менее жесткую пружину `\hfil`.

Разберем действие пружин на следующих примерах.

Пример 1: `\hfil \tiger \hfil \tiger \\`


Здесь картинки расположены в строке несимметрично. Первая из них окружена пружинами `\hfil`, из которых левая, находящаяся в начале строки, игнорируется как обычный пробел. Вторая картинка находится на одинаковом расстоянии от правой границы текста и первой картинки, так как она окружена одинаковыми пружинами, одна из которых задана явно командой `\hfil`, а вторая неявно стоит в конце строки.

Пример 2: `\mbox{}\hfil \tiger \hfil \tiger \\`


В данном примере в начало строки поставлен пустой бокс `\mbox{}`, служащий «упором» для первой пружины. Как следствие, обе картинки,

[1] Команда `\tiger[•]` определена как `\includegraphics[width=•]{tiger}`.

оказавшись зажатыми одинаковыми пружинами, распределились по строке симметрично.

Пример 3: `\mbox{}\hfil \tiger \hfill \tiger \\`


В последнем примере между картинками поставлена жесткая пружина `\hfill`, смявшая пружины `\hfil`, находящиеся в начале и конце строки, поэтому картинки разошлись по краям страницы.

Пара жестких пружин `\dotfill` и `\hrulefill` заполняют пробел точками или линией, например:

```
|\dotfill|\hrulefill|\\
```

|.....|_____|

2.3. Выделение и выравнивание текста

Часть текста можно выделить подчеркиванием или отбивками, изменив шрифт или метод выравнивания, а также изменив цвет или фон текста. *Отбивкой* называется дополнительное пустое поле вокруг или с какой-либо стороны объекта. Использование шрифтов обсуждается в разд. 2.6 и гл. 12, а цветная верстка — в разд. 7.7.

Для подчеркивания LaTeX имеет команду `\underline{`*текст*`}`, создающую бокс, в который помещается часть строки, например `\underline{слово}` ⇝ <u>слово</u>. Дополнительные возможности, включающие подчеркивание нескольких строк с использованием различных линий, а также вычеркивание и штриховка обсуждаются в разд. 7.6.

Для выделения текста предназначена пара окружений `quote` и `quotation`. Помещенный в них текст отбивается со всех сторон пустыми полями, хорошо выделяющими его на странице.

> В окружении `quote` абзацы разделяются небольшой вертикальной отбивкой и не имеют отступа. Это видно в данном примере, в котором шрифт уменьшен для наглядности.
>
> Выделение хорошо смотрится, если абзацы достаточно велики. Не стоит применять его для отдельных строк, так как они будут выглядеть очень рыхлыми.

Чтобы показать разницу в оформлении текста этими окружениями, другой пример поместим в окружение quotation. Отметим, что

текст обоих примеров содержит полезные комментарии об особенностях верстки абзацев.

> В окружении quotation абзацы начинаются с отступа, а отбивка между ними не делается. В данном примере также для наглядности использован уменьшенный шрифт.
>
> Текст верстается в колонку, ширина которой меньше ширины окружающего текста. Как и в случае quote, для хорошего зрительного выделения каждый абзац должен содержать по несколько строк, чтобы границы текста были хорошо очерчены.

Изменение метода выравнивания текста отключает таблицу переноса слов и подавляет отступы. В добавок к этому, в каждую строку неявно вставляются пружины `\hfil`: текст, выравниваемый по левому краю, поджимается пружинами справа; в тексте, выровненном по правому краю, пружины стоят в начале строк; центрированный текст сжимается пружинами, стоящими по его краям.

Для выравнивания абзацев и больших массивов текста применяются декларации и окружения, а для отдельных строк команды, представленные в табл. 2.4.

Таблица 2.4

Средства выравнивания текста

Действия	Окружения	Декларации	Команды
Центрирование	center	`\centering`	`\centerline{•}`
Выравнивание слева	flushleft	`\raggedright`	`\leftline{•}`
Выравнивание справа	flushright	`\raggedleft`	`\rightline{•}`

Выравниваемая строка целиком помещается в аргумент одной из команд. Если она составляет часть абзаца,`\newline`

`\rightline{`как в данном примере,`}\newline`

предшествующую строку нужно завершить командами `\\`, `\newline` или `\linebreak`, а ее закончить командами `\\` или `\newline`, как показано в примере. В случае отдельно стоящей строки команды разрыва строк не требуются.

Чаще всего декларации `\centering`, `\raggedleft` и `\raggedright` используют для выравнивания рисунков и таблиц внутри окружений figure или table. Текст лучше выравнивать, заключив его в окружение minipage, бокс `\parbox` или просто в группирующие скобки. В окружениях и боксах декларации действуют автоматически, а в группе, созданной скобками, их действие нужно инициировать командой `\\`, поставив ее в конце. Команда `\newline` в данном случае не работает.

Декларации и окружения верстают текст немного по-разному.

```
\begin{flushleft}
```
Сверху и снизу текста, выровненного окружениями, делается небольшая отбивка. Чтобы показать это, данный абзац помещен в окружение flushleft. Как видно из этого примера, слова не переносятся, поэтому правая граница текста становится «рваной».
```
\end{flushleft}
```

{\raggedleft Этот абзац, для примера выровненный декларацией \raggedleft, демонстрирует, что текст, выравниваемый таким образом, не отбивается. Его верхнюю отбивку создает окружение flushleft, а снизу отбивки нет.\par}

Строка следующего абзаца, подтверждающая отсутствие отбивки.

2.4. Вертикальное выравнивание

Из абзацев формируются страницы. Распределяя текст по страницам, компилятор оптимизирует промежутки между абзацами, формулами, заголовками, таблицами, рисунками и и т. п.. Единица вертикального выравнивания текста — *абзац*, но и внутри него расстояние между строками может меняться.

Величина межстрочного интервала определяется произведением коэффициента `\baselinestretch` на длину `\baselineskip`, зависящую от размера шрифта. Команда `\baselinestretch` хранит десятичное число, регулирующее расстояние между строками. По умолчанию его значение равно единице, но если его изменить, например в полтора раза (см. разд. 7.1), `\renewcommand\baselinestretch{1.5}`, межстрочный интервал пропорционально изменится. Если же задать значение меньше единицы или нуля, строки не «наедут» друг на друга, так как расстояние между ними ограничивается снизу суммой глубины предыдущей строки и высоты следующей.

Как говорилось в разд. 2.2.1, внутри абзаца расстояние между строками позволяет регулировать команда перевода строки. Ее вариант «со звездочкой»:

`\\*[`*длина*`]`,

разрывая строку, следит, чтобы следующая оказалась с ней на одной странице. Напомним, что при положительной *длине* следующая строка сдвигается вниз, а при отрицательной — вверх.

Команды вертикальных пробелов, сведенные в табл. 2.5, вставляют промежутки между абзацами. Обычно их ставят в начале или в конце абзацев. Команды, хранящие длины пробелов, указаны во второй колонке, а в третьей приведены их значения по умолчанию. Вдобавок к командам, формирующим промежутки заданной высоты, имеются команды `\vspace`, аналогичные горизонтальным пробелам, позволяющие регулировать высоту промежутка по своему усмотрению. Вертикальные пробелы, попадающие в начало или конец страницы, игнорируются, но сдвиг, заданный командой `\vspace*`, выполняется всегда.

Таблица 2.5

Команды вертикальных пробелов

Команды	Длины	Значения
`\smallskip`, `\medskip`, `\bigskip`, `\vspace{`*длина*`}`, `\vspace*{`*длина*`}`.	`\smallskipamount`, `\medskipamount`, `\bigskipamount`,	3pt ± 1p, 6pt ± 2pt, 12pt ± 4pt,

Наряду с пробелами существуют и вертикальные пружины `\vfil` и `\vfill`, вставляемые между абзацами. Находясь внутри абзаца, пружины автоматически разбивают его, а оказавшись же в начале или конце страницы, они, как и другие пробелы, игнорируются. По сравнению с `\vfil` пружина `\vfill` обладает большей жесткостью:

```
\mbox{}\vfil

Размер страницы показан рамкой.
```

```
\mbox{}\vfill

Размер страницы показан рамкой.
```

В конце страницы LaTeX неявно использует в пружину `\vfil`, прижимающую текст вверх. Это легко проверить, поставив в начале новой страницы конструкцию `\mbox{}\vfil`, как показано в примере. В этом случае пустой бокс послужит упором для пружины, и набираемый текст, оказавшись между двумя одинаковыми пружинами, сдвинется в середину страницы. Аналогичная конструкция с командой `\vfill` приведет к тому, что текст опустится вниз, так как более жесткая пружина сомнет более мягкую. Используя аналогию выравнивания

строки пружинами \hfil и \hfill, легко построить нужный шаблон вертикального выравнивания текста на странице.

Регулировать разбиение текста на страницы помогают команды

\newpage, \pagebreak[число], \nopagebreak[число].

Команда \newpage, как следует из ее имени, должна начать новую страницу или новую колонку при многоколоночном наборе. Она не заботится о равномерности распределении текста по высоте разрываемой страницы. Если страница окажется слишком пустой, компилятор может проигнорировать ее выполнение.

Команда \pagebreak также завершает текущую страницу и начинает новую, а команда \nopagebreak, напротив, препятствует переходу на следующую страницу. Их параметр позволяет управлять жесткостью предписываемого компилятору указания от рекомендации (0), до обязательного выполнения (4), используемого по умолчанию. Команда \nopagebreak учитывается только при автоматическом разбиении текста на страницы, если же какая-то другая команда явно инициирует разрыв страницы, она игнорируется.

В отличие от команды \newpage, не заботящейся о выравнивании разрываемой страницы, \pagebreak может это сделать. Выравнивание зависит от действия деклараций \flushbottom и \raggedbottom. Если активна декларация \flushbottom, страницы заполняются сверху до низу за счет увеличения промежутков между абзацами, как сделано в этой книге. Если выравнивание отменено декларацией \raggedbottom, внизу страниц остается пустое поле.

Эти декларации являются глобальными и их использование меняет режим верстки всех последующих страниц. В стандартных классах выравнивание \flushbottom активируется при использовании параметра twoside, чтобы обе страницы разворота были одинаковы. Для параметра oneside вертикальное выравнивание не используется.

Компилятор стремится не допустить переполнение страниц, поэтому команда \nopagebreak эффективна не всегда. В таких случаях можно изменить высоту текста страницы, используя команды

\enlargethispage{длина} и \enlargethispage*{длина},

увеличивающие его на указанную длину. Последняя дополнительно минимизирует имеющиеся на странице вертикальные пробелы.

В качестве примера использования команд, регулирующих вертикальное выравнивание, разберем код титульной страницы отчета.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
Национальный исследовательский ядерный университет «МИФИ»
(НИЯУ МИФИ)

Отчет о научно-исследовательской работе
«Современное состояние проблемы массы нейтрино»

Работу выполнил
магистр гр. ●-● И. И. Иванов

Руководитель
доц. кафедры № ● П. П. Петров

2024-08-10

```
\newpage
\thispagestyle{empty}
\begin{center}

   МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ \\
  \mbox{} \\
   ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
       ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ \\
   Национальный исследовательский ядерный университет «МИФИ» \\
   (НИЯУ МИФИ)

\vfill
   Отчет о научно-исследовательской работе             \\
   <<Современное состояние проблемы массы нейтрино>>
\vfill

\mbox{}\hfill
  \begin{tabular}{lp{5em}r}
      Работу выполнил  & & \\
      магистр гр. ●-● & & И.И. Иванов\\[1ex]
      Руководитель       & & \\
      доц. кафедры № ● && П.П. Петров
  \end{tabular}

\vfill
\today

\end{center}
\newpage
```

Команды `\newpage`, стоящие в начале и конце примера, ограничивают текст верстаемой страницы, а команда `\thispagestyle{empty}` подавляет вывод ее номера (см. разд. 8.9.1).

Для центрирования текста использовано окружение center.

Наименования министерства и университета разделены строкой с пустым боксом `\mbox{}`. Без него строка будет пустой и попытка ее разрыва командой `\\` даст ошибку компиляции. Остальные объекты разделены пружинами `\vfill`, обеспечивающими их равномерное распределение по высоте страницы.

Сведения об исполнителе и руководителе оформлены в виде таблицы, прижатой к правому краю страницы пружиной `\hfill`, упором для которой служит пустой бокс. Подписи исполнителя и руководите-

ля разделены небольшим вертикальным пробелом путем раздвигания строк с помощью команды `\\[1ex]`. Команда `\today` выводит дату создания отчета.

2.5. Министраницы

Возможности верстки значительно расширяют министраницы. При их создании достаточно объявить их ширину, зная которую, компилятор верстает их как обычные страницы, нарезая текст на строки заданной длины. Сверстанная министраница помещается в бокс, который встраивается в строку на общих основаниях.

Министраницы можно использовать в любом месте рукописи вместо обычного символа текста. Они могут содержать сноски и практически любые объекты, кроме плавающих (см. разд. 3.3).

Министраницу создает окружение minipage :

`\begin{minipage}[`*привязка*`][`*высота*`][`*выравнивание*`]{`*ширина*`}` ,

имеющее аргумент, задающий ширину министраницы, и набор параметров, определяющих ее высоту, а также методы встраивания в строку и распределения текста по высоте.

Привязка к строке задается следующими идентификаторами:

- **c** — министраница центрируется относительно базисной линии строки, в которую она встраивается;
- t — верхняя строка министраницы встраивается в строку;
- b — в строку встраивается нижняя строка министраницы.

Если задана высота министраницы, текст можно распределить по вертикали с помощью идентификаторов c, b, t или s:

- c — текст центрируется;
- b — текст прижимается к нижнему краю;
- t — текст прижимается к верхнему краю;
- s — текст равномерно распределяется по высоте министраницы.

По умолчанию министраница центрируется в строке, а для распределения текста применяется метод выравнивания в строке.

Аналогом окружения minipage является команда:

`\parbox[`*привязка*`][`*высота*`][`*выравнивание*`]{`*ширина*`}{`*текст*`}` ,

имеющая такой же синтаксис и тоже создающая министраницу. Ее удобно использовать для небольших фрагментов текста.

Министраницы многократно использованы в книге для компоновки примеров, имеющих следующую структуру:

```
              \begin{minipage}{•}
                \begin{verbatim}
\mbox{}\hfil     Текст рукописи.  \hfil  \parbox{•}{Текст документа.}
                \end{verbatim}
              \end{minipage}
```

Перед первой министраницей, содержащей текст рукописи, стоит пружина с упором, и еще одна между министраницами. Вся конструкция находится в одной строке, поэтому пружины создают симметричные отбивки по бокам министраниц, а общая привязка к строке центрирует их друг относительно друга по вертикали.

2.6. Шрифты

Важной составляющей верстки является шрифтовое оформление документа. Следует учитывать, чем реже меняется шрифт, тем легче читается текст, а хорошо подобранные шрифты названий разделов, подписей, сносок и т. д. выделяют их из основной массы текста и делают наглядной структуру документа.

По умолчанию, стандартные классы LaTeX подключают шрифты Computer Modern, использованные в верстке данной книги.

Текстовый шрифт имеет следующий набор атрибутов: гарнитура (семейство), контрастность, начертание, *кегль* (размер) и кодировка. Любой из них можно изменять независимо от других, при этом последовательность изменения нескольких атрибутов не влияет на конечный результат.

2.6.1. Гарнитуры Computer Modern

Гарнитура — совокупность шрифтов разной насыщенности, с различным начертанием и размером, принадлежащих одному семейству.

Шрифты делятся на пропорциональные, литеры которых имеют разную ширину, и моноширинные, у которых ширина литер одинакова. Пропорциональные шрифты, в свою очередь делятся на шрифты «с засечками», придающими словам слитность, и рубленые, не имеющие засечек. Стандартный набор гарнитур включает в себя пропорциональные шрифты Serif и SansSerif и моноширинный шрифт Mono.

Гарнитуры Serif, используемые наиболее часто, представляют собой пропорциональные шрифты с засечками.

Гарнитуры SansSerif или просто Sans относятся к категории пропорциональных рубленых шрифтов, подобных шрифту, которым оформлен данный абзац.

Гарнитуры Mono содержат шрифты с фиксированной шириной литер. В зависимости от дизайна они могут быть как рубленными, так и с засечками.

Шрифты Computer Modern содержат три гарнитуры: Roman (Serif), SansSerif и TypeWriter (Mono).

Контрастность шрифта определяют ширина символов и насыщенность, зависящая от их «толщины». Шрифты бывают светлые (тонкие), нормальные, жирные и т. п. Качественные типографские шрифты имеют большой диапазон контрастностей. Гарнитуры Computer Modern содержат нормальные и **уширенные жирные шрифты**.

Начертание определяет стиль формы символов. В шрифтах Computer Modern имеется прямое начертание, *наклонный шрифт, курсив, или italic,* капитель, в которой строчные буквы имеют вид уменьшенных прописных, и прямой курсив, использованный нами в описаниях команд.

Гарнитура Roman наиболее богата шрифтами, имеющими нормальную и жирную насыщенность всех перечисленных начертаний, кроме прямого курсива:

Обычный шрифт (upright).
Курсив (italic).
Прямой курсив (upright italic).
Прописной шрифт или капитель (Small Capitals).
Наклонный шрифт (slanted).
Прописной шрифт или капитель (Small Capitals).
Жирный шрифт (bold).
Жирный курсив (bold italic).
Жирный наклонный шрифт (slanted bold).

Гарнитура SansSerif содержит шрифт нормальной насыщенности и жирный шрифт. Начертания наклонного шрифта и курсива совпадают, капитель подставляется из гарнитуры Roman, а жирной капители нет.

Обычный шрифт (upright).
Наклонный шрифт и курсив (slanted ≡ italic).
Жирный шрифт (bold).
Жирный курсив (bold, slanted ≡ italic).

```
В гарнитуре TypeWriter имеются шрифты нормальной
насыщенности всех начертаний. Жирные шрифты отсутствуют.
В тексте, набранном в этой гарнитуре, автоматический перенос
слов не используется, пробелы имеют фиксированную ширину,
а разбивку на строки нужно регулировать вручную командами \\.
```

Обычный шрифт (upright).
Курсив (italic).
Наклонный шрифт (slanted).
Пример Капители

2.6.2. Атрибуты шрифтов

Помимо перечисленных выше начертаний, гарнитуры Serif могут содержать вычурные шрифты с каллиграфическими заглавными буквами и разреженную капитель, в которой между буквами добавлен небольшой промежуток, делающий ее более гармоничной. Разреженная капитель бывает также и в SansSerif гарнитурах. Примеры таких шрифтов показаны на рис. 12.7 – 12.9.

Атрибутами шрифтов управляет набор команд, деклараций и окружений, представленных в табл. 2.6. Команды предназначены для шрифтового оформления небольших фрагментов текста в пределах абзаца, а окружения — для больших массивов текста. Декларации удобно применять в группах, например ячейках таблиц, заголовках, подписях, аргументах и параметрах команд и т. д., так как группа, автоматически ограничивает область изменения шрифта.

Как уже отмечалось, атрибуты шрифта могут комбинироваться в любом порядке, поэтому последовательности команд
`\textbf{\itshape •}`, `\textbf{\textit{•}}`, `{\bfseries\itshape •}`, `\textit{\bfseries •}`, `\textit{\textbf{•}}`, `{\itshape\bfseries •}` эквивалентны. Все они печатают текст ***жирным курсивом***.

Прямое, курсивное и наклонное начертания конкурируют друг с другом, тогда как капитель комбинируется с ними. Например, декларация `\slshape` *заменит* курсивный или прямой шрифт наклонным, а капитель, если в гарнитуре есть наклонная капитель, *наклонит*.

Команда `\emph{•}` и декларация `\em` заменяют прямой шрифт курсивом, и наоборот, в зависимости от того, какое начертание имел предшествующий текст: `\emph{` *Их действие иллюстрирует* `\em\` данный пример. `}`

Таблица 2.6

Команды, изменяющие атрибуты стандартных гарнитур

Команды	Декларации	Окружения	Примеры гарнитур		
`\textrm{•}`	`\rmfamily`	rmfamily	Serif		
`\textsf{•}`	`\sffamily`	sffamily		SansSerif	
`\texttt{•}`	`\ttfamily`	ttfamily			Mono
`\textmd{•}`	`\mdseries`	mdseries	Medium	Medium	Medium
`\textup{•}`	`\upshape`	upshape	Upright	Upright	Upright
`\textit{•}`	`\itshape`	itshape	*Italic*	*Italic*	*Italic*
`\textsl{•}`	`\slshape`	slshape	*Slanted*	*Slanted*	*Slanted*
`\textsw{•}`	`\swshape`	swshape	*Swash*	Swash	Swash
`\textsc{•}`	`\scshape`	scshape	SmallCapitals	SmallCapitals	SmallCapitals
`\textssc{•}`	`\sscshape`	sscshape	SmallCapitals	SmallCapitals	SmallCapitals
`\textbf{•}`	`\bfseries`	bfseries	**Bold**	**Bold**	Bold
`\textup{•}`	`\upshape`	upshape	**Upright**	**Upright**	Upright
`\textit{•}`	`\itshape`	itshape	***Italic***	***Italic***	*Italic*
`\textsl{•}`	`\slshape`	slshape	***Slanted***	***Slanted***	*Slanted*
`\textsw{•}`	`\swshape`	swshape	***Swash***	**Swash**	Swash
`\textsc{•}`	`\scshape`	scshape	**SmallCapitals**	**SmallCapitals**	SmallCapitals
`\textssc{•}`	`\sscshape`	sscshape	**SmallCapitals**	**SmallCapitals**	SmallCapitals
`\emph{•}`	`\em`	em	*Italic* ⇆ Upright	*Italic* ⇆ Upright	*Italic* ⇆ Upr.

Команда `\textulc{•}` и декларация `\ulcshape` переводят капитель в строчные буквы, сохраняя начертание, с которым она комбинировалась. Например, {\slshape\scshape *наклонную капитель декларация* \ulcshape\ *сделает наклонным шрифтом*}, \textsc{а прямая капитель \textulc{станет прямым шрифтом}}. Декларация `\normalshape` возвращает шрифту начертание, используемое по умолчанию (обычно прямое). Декларация `\normalfont` активирует основной шрифт документа (см. разд. 12.2), устанавливая все его атрибуты: кодировку, семейство, контрастность и начертание. Например, \sffamily\bfseries **жирный рубленный шрифт после ее применения,** \normalfont станет обычным шрифтом.

Конструкции, изменяющие несколько атрибутов шрифта, оказываются довольно громоздкими, поэтому для быстрого доступа к часто используемым шрифтам определены специальные декларации:

`\sf`	`= \sffamily\mdseries\upshape`	Стандартный SansSerif.
`\tt`	`= \ttfamily\mdseries\upshape`	`Стандартный Mono`
`\rm`	`= \rmfamily\mdseries\upshape`	Стандартный Serif.
`\sc`	`= \rmfamily\mdseries\scshape`	Капитель Serif.
`\it`	`= \•family\mdseries\itshape`	*Курсив Serif.*
`\sl`	`= \•family\mdseries\slshape`	*Наклонный Serif.*
`\bf`	`= \•family\bfseries\upshape`	**Жирный Serif.**

Здесь и далее знак равенства означает не буквальность воспроизведения кода, а идентичность действия команд. Декларация, обозначенная `\•family`, устанавливает семейство основного шрифта. Обычно он принадлежит семейству Serif и `\•family=\rmfamily`, однако если основным семейством является SansSerif (см. разд. 12.2.2), то `\•family=\sffamily` и {\it *соответствует наклонному рубленому шрифту, а* \bf — **жирному рубленому шрифту**. }

Покажем действие коротких команд, изменив для примера оформление заголовков раздела:

\section*{Заголовок},
\section*{\sf Заголовок},
\section*{\sc Заголовок}.

2.6.3. Размеры шрифтов

Размеры шрифтов различных составляющих частей документа отсчитываются от кегля основного шрифта, устанавливаемого при загруз-

ке класса документа. В стандартных классах для него определены три размера: 10pt, 11pt и 12pt. Коллекция пакетов extsizes [18] добавляет к ним размеры 8pt, 9pt, 14pt, 17pt и 20pt. Чтобы их использовать, нужно загрузить класс документа, начинающийся с префикса ext: extarticle, extbook, extletter, extproc, extreport.

Размера шрифта изменяют декларации и окружения, перечисленные в табл. 2.7. Их имена одинаковы, и чтобы не загромождать таблицу, окружения указаны не для всех размеров. Пары значений кегль/(межстрочный интервал) приведены в единицах pt. Примеры показаны для основного размера 10pt. Некоторые значения округлены, точные величины равны 10.95pt, 14.4pt, 17.28pt, 20.74pt, 24.88pt, 29.86pt, 35.83pt.

Таблица 2.7

Градации размеров шрифтов

Декларации, окружения[2]	Примеры	10pt	11pt	12pt	14pt
\tiny, tiny	Abcdefghijkl	5/6	6/7	6/7	6/7
\scriptsize	Abcdefghij	7/8	8/9.5	8/9.5	8/9.5
\footnotesize	Abcdefghi	8/9.5	9/11	10/12	10/12
\small, small	Abcdefgh	9/11	10/12	11/13.6	12/14
\normalsize	Abcdefg	10/12	11/13.6	12/14.5	14/17
\large, large	Abcdefg	12/14	12/14	14/18	17/22
\Large, Large	Abcdef	14/18	14/18	17/22	21/25
\LARGE, LARGE	Abcd	17/22	17/22	21/25	25/30
\huge, huge	Abcd	21/25	21/25	25/30	30/35
\Huge, Huge	Abc	25/30	25/30	25/30	36/40

Область действия деклараций ограничена группами, поэтому изменение кегля внутри аннотации, заголовков, подписей, сносок и т. д. не затрагивает текст за их пределами.

Чтобы тест не был слишком разреженным или плотным, в окружениях при смене кегля меняется также и длина \baselineskip, задающая межстрочный интервал. При использовании деклараций меняется только кегль, а чтобы изменить межстрочный интервал, текст следует

[2] Чтобы не загромождать таблицу, окружения scriptsize, footnotesize и normalsize не указаны.

поместить в отдельный абзац (см. разд. 12.2.7) или на министраницу (см. разд. 2.5).

Пример уменьшения шрифта заголовка приведен ранее, а здесь скажем всем разработчикам LaTeX

```
{\Huge «огромное спасибо»! }
```

{\Huge «огромное спасибо»! }

2.6.4. Кодировка текста и шрифтов

Кодировка рукописи и набор используемых шрифтов определяют совокупность символов, доступных при верстке. Чтобы символ попал в печатный документ, компилятор должен определить его привязку к шрифтам. Компилятор pdflatex оперирует несколькими типами кодировок. По умолчанию файлы рукописи имеют кодировку, которая является полновесной реализацией кодировки unicode, UTF8, охватывающей десятки тысяч символов. Печатные шрифты, содержащие *глифы*, т. е. буквы и символы, которыми печатается сверстанный документ, как правило, тоже имеют кодировку unicode. В то же время pdflatex работает только с 8-битными кодировками, вмещающими лишь 256 символов (кодов), поэтому каждый печатный шрифт разбивается на множество 8-битных кодировок. Но при этом количество символов, которыми оперирует компилятор не ограничено, так как в начале верстки они преобразуются в команды, а в ее конце команды преобразуются в глифы печатных шрифтов. Первое преобразование настраивает пакет inputenc [19],

```
\usepackage[кодировка]{inputenc},
```

загружающий *кодировку*, связывающую символы набираемого текста с соответствующими командами, например: Б ⇝ `\IeC {\CYRB }` или æ ⇝ `\ae`. Она должна соответствовать кодировке файлов рукописи, иначе после компиляции документ будет содержать абракадабру.

Полный набор букв латиницы и кириллицы обеспечивает кодировка utf8, позволяющая набирать документы на всех европейских языках, кроме греческого. Поддерживаемый ею набор символов, представленный в табл. 2.8, весьма ограничен по сравнению с полной кодировкой UTF8, но в подавляющем большинстве случаев его вполне достаточно.

Чтобы в сверстанном документе команды преобразовались в глифы шрифтов, они должны входить в загружаемые 8-битные кодировки, коды (символы) которых однозначно связаны с глифами. Ядро LaTeX

Таблица 2.8

Буквы и символы кодировки utf8

Символы кодировок T1, TS1 и T2A

0	1	2	3	4	5	6	7	8	9	#
{	}	{	}	⟦	⟧	$	$	₩	₪	
\	†	%	—	¡	⊙	◉	¶	¢	₲	√
/	†	‰	–	⁉	©	◯	¶	¢	₿	ƀ
¦	‡	‰	—	¿	℗	∘	§	₡	₫	₠
‖	★	*	–	¡	®	•	№	f	£	€
⟨	⟩	«	»	...	SM	°	℃	₦	₽	₨
↑	↓	‹	›	·	TM	⁒	¤	€	₩	¥
←	→	a	o	1	2	3	½	¼	¾	♪
␣	±	×	÷	※	¬	▪	µ	Ω	℧	
˝	˜	˜	¯	¨	´	″	ˆ	ˇ	˘	

Буквы латиницы

Àà	Áá	Ââ	Ăă	Ăă	Ää	Åå	Ãã	Āā	Ąą	**Ạạ**
Ææ	Ǣǣ	Ḃḃ	**Ḅḅ**	Ćć	Ċċ	Ĉĉ	Čč	Çç	Ďď	**Ḍḍ**
Ðð	Đđ	Èè	Éé	Êê	Ĕĕ	Ěě	Ëë	Ėė	Ēē	Ęę
Ḛḛ	Ǵǵ	Ĝĝ	Ğğ	Ǧǧ	Ġġ	Ḡḡ	Ģg	**Ģg**	Ĥĥ	**Ḥḥ**
Ħħ	Ìì	Íí	Îî	Ǐǐ	Ĭĭ	İı	Ïï	Ĩĩ	Īī	Įį
Ịị	IJ ij	Ĵĵ	ǰ	ȷ	Ǩǩ	Ķķ	**Ḳḳ**	Ĺĺ	Ľľ	Ļļ
Ḷḷ	Łł	**Ṃṃ**	Ńń	Ňň	Ññ	Ņņ	**Ṇṇ**	**Ŋŋ**	Òò	Óó
Ôô	Ǒǒ	Ŏŏ	Ő̋ő̋	Ōō	Õõ	Öö	Ǫǫ	**Ǫǫ**	Øø	Œœ
Ŕŕ	Řř	Ŗŗ	**Ṛṛ**	Śś	Ŝŝ	Šš	Şş	**Ṣṣ**	ß ß	
Ťť	Ţţ	Ṭṭ	**Ṭṭ**	**Þþ**	Ùù	Úú	Ûû	Ŭŭ	Ǔǔ	Ůů
Üü	Ű̋ű̋	Ũũ	Ūū	Ųų	**Ụụ**	**Ṿṿ**	Ŵŵ	**Ẉẉ**	Ýý	Ŷŷ
		Ÿÿ	Ȳȳ	**Ỿỿ**	Źź	Žž	Żż	**Ẓẓ**		

Буквы кириллицы

Ăă	Ää	Ææ	Ѓѓ	Ґґ	Ғғ	**Ҕҕ**	**Ґ̣ґ̣**	**Ғ̣ғ̣**	Ѐѐ	Ёё
Ёё	Ӝӝ	Ӝӝ	Җҗ	Ӟӟ	Ҙҙ	Йй	Ӣӣ	Йй	Йй	Ќќ
Ққ	Кк	**Ҟк**	Ҡҡ	**Ҟ̣ҟ̣**	Љљ	**Л̣л̣**	**М̣м̣**	Ңң	Ҥҥ	Њњ
Ңң	Ӧӧ	Өө	**Ҧҧ**	**Ҏҏ**	Ҫҫ	**Тҭ**	**Ц̣ц̣**	Ӱӱ	Ӯӯ	Ӳӳ
Ӱӱ	Хх	**Ҳҳ**	Ҳҳ	Ҷҷ	Чч	**Ч̣ч̣**	Ҹҹ	Ыы	**Ъъ**	Ӛӛ
Єє	Іі	Її	Јј	Ѕѕ	Уу	Ӯӯ	Ҥҥ	Ђђ	Ћћ	Џџ
Əə	Ӛӛ	**Ҽҽ**	Ҿҿ	**Ҩҩ**	**Ҙҙ**	**Ъ̣ъ̣**	**Ӂ̣ӂ̣**	**Өө**	**Vv**	**V̈v̈**

загружает кодировку T1, обеспечивающую поддержку ASCII-символов и букв латиницы (см. табл. Б.1) и кодировку TS1, содержащую небуквенные символы (см. табл. Б.2). Для русского языка пакет babel загружает и активирует кодировку T2A (см. табл. Б.1). Таким образом, для набора русского текста достаточно загрузить кодировку utf8, пакет cmap и настройки русского языка:

```
\usepackage[utf8]{inputenc}
\usepackage{cmap}
\usepackage[russian]{babel}
```

Такая преамбула неявно подключает шрифты латиницы и кириллицы. Работая с текстом, набранным латиницей, можно ограничиться лишь первой командой.

Текст сверстанного pdf-документа можно копировать в другие приложения, но в программе AdobeReader с этим могут возникнуть проблемы. Чтобы избежать их, пакет cmap добавляет в документ привязку 8-битных кодировок к unicode. Его следует загружать *до* пакета babel. Более подробно 8-битные кодировки рассмотрены в прил. Б.

В табл. 2.8 собраны символы и буквы кодировки utf8. Для наглядности они представлены без печатающих их команд, которые приведены в табл. Б.3, Б.5 и Б.6. Символы, представленные в трех первых строках, и буквы, выделенные жирным шрифтом, вводятся в виде команд, а остальные в «натуральном виде». Используйте для этого ресурсы, описанные на с. 13.

Таблица 2.9

Диакритические знаки

ò	ó	ö	ô	õ	ō	ȯ	ȍ	ő	o̊	ô	ǒ	ŏ	ǒ	o͡o
					o̲	o̧	o̩	o̤	o̩					

Многие алфавиты содержат буквы с диакритическими знаками, или акцентами, проставляемыми над или под основной буквой, например буквы «ё» и «й» в русском алфавите. Большинство акцентированных букв охвачены кодировками T1 и T2A. В рукописи их можно использовать как обычно, а для печати букв, не входящих в данные кодировки, нужно пользоваться командами LaTeX, позволяющими акцентировать любую букву. Полный набор диакритических знаков приведен в табл. 2.9 на примере буквы «о». Генерирующие их их команды представлены в табл. Б.4

Хотя рукопись набирается по-русски, во время компиляции русские буквы заменяются командами `\IeC{\CYR• }`. Например, имя Петр пре-

образуется к виду `\IeC{\CYRP}\IeC{\cyre}\IeC{\cyrt} \IeC{\cyrr}`. Такой вид могут приобрести ярлыки и метки, направляемые в служебные файлы. Другие программы такую информацию не воспринимают, и при компиляции документа, включающей генерацию оглавления, списка литературы, предметного указателя и т. д., это может вызвать ошибки. Начиная с 2020 г., ядро LaTeX печатает текст служебных файлов буквами, однако некоторые пакеты, в том числе и стандартные, пока еще могут работать по старому алгоритму, и появление команд в служебных файлах все еще возможно, поэтому использование русских букв в ярлыках и метках связано с риском.

Еще одним источником ошибок, вызванных кодировками, является копирование текста из других приложений. В нем могут находиться символы, не входящие в кодировку utf8, внешне неотличимые от стандартных, например специальные пробелы вместо обычных, или математический минус вместо тире. Они вызывают ошибки со следующими сообщениями:

```
! Package inputenc Error: Unicode character •(U+####)
(inputenc)    not set up for use with LaTeX,
```

Вместо • и #### в сообщениях находятся символы и их шестнадцатеричные коды. При появлении такой ошибки нужно заменить указанный символ командой LaTeX или аналогичным стандартным символом. В TeXstudio это делается следующим способом. По умолчанию в окне сообщений, расположенном под окном редактирования, активируется закладка «Log», содержащая краткую сводку сообщений компиляторов, подготовленную самим TeXstudio. Чтобы увидеть оригинальное сообщение, откройте закладку «Log File», тогда в нижней части окна появится приведенное выше описание ошибки. Скопируйте из него символ, вызвавший ошибку, в меню «Edit ⇒ Searching ⇒ Replace» и замените его по всему документу.

2.7. Цифры, тире, кавычки, точки, запятые

В LaTeX определено большое количество различных кавычек. Их можно вводить с помощью команд и с клавиатуры, как обычные символы. Для комфортного набора кавычек в шрифтах также предусмотрены лигатуры. *Лигатура* — преобразование последовательности из нескольких символов в общий символ. Сравните, например, последовательность букв `f{}f{}l` ⇝ ffl с их лигатурой `ffl` ⇝ ffl.

Две запятые дают лигатуру нижней кавычки (`,,` ⇝ „), две верхние левые и правые кавычки преобразуются в двойные верхние кавычки (`''` ⇝ ” и ``` `` ``` ⇝ “). В англоязычной литературе используются одинарные и двойные верхние кавычки, при этом одинарные «вкладываются» в двойные: “‘To be, or not to be?’ — that is the question.”

В русской полиграфии используются „лапки“ и «елочки», для которых также есть лигатуры: `<<` ⇝ «, `>>` ⇝ ». «Следует помнить, что основными кавычками являются „елочки“, а „лапки“, как правило, служат в качестве вложенных!». Для елочек пакет babel вводит дополнительные команды `"<` ⇝ « и `">` ⇝ ».

Дефис, используемый для разделения слов (например, *темно-красный*), вводится обычным образом с клавиатуры. Для ввода тире имеются команды и лигатуры. Короткое тире, применяемое для разделения диапазона чисел, вводится с помощью лигатуры `--` или команды `\textendash`. Например, `7--8` ⇝ 7–8, а лучше `7\,--\,8` ⇝ 7 – 8. Тот же диапазон, набранный в формуле как вычитание чисел, выглядит иначе: `$7-8$` ⇝ $7 - 8$. Длинное тире, используемое в сложных предложениях, вводится с помощью лигатуры `---` или команды `\textemdash`: `<<Зачем тебе жужжать, если ты не пчела?>> --- подумал Винни Пух.` ⇝ «Зачем тебе жужжать, если ты не пчела?» — подумал Винни Пух.

Для русского языка пакет babel определяет целых три типа длинных тире. Одно из них, `"--~` ⇝ — эквивалентно лигатуре `---` или команде `\textemdash`. Оно не имеет отбивок, а два других выводятся с небольшой правой отбивкой: `"---` ⇝ |—| и `"--*` ⇝ |— |. Последний вариант рекомендован для печати прямой речи. В примерах тире с отбивками для наглядности показаны между вертикальными палочками.

Команда `\dots`, определенная как в тексте, так и в формулах, печатает многоточия, например `$\pi=3.14\dots$` ⇝ $\pi = 3.14\ldots$ и т. п…

В англоязычных странах разделителем целой и дробной частей числа служит десятичная точка. В российской полиграфии для этого чаще используется запятая, однако это не является обязательным требованием. В документах, верстаемых в LaTeX, следует пользоваться десятичной точкой, так как в формулах запятая автоматически отделяется небольшим пробелом, разбивающим число на две части. Сравните $\pi = 3.14159\ldots$, $\mathrm{e} = 2.71828\ldots$ и $\pi = 3,14159\ldots$, $\mathrm{e} = 2,71828\ldots$ Из-за этого зарубежные издательства и редакции переводных российских журналов не принимают рукописи с десятичной запятой.

Шрифты Computer Modern, по умолчанию используемые LaTeX, содержат два варианта арабских цифр — в обычном и старом стилях. Для печати цифр старого стиля используется команда

`\oldstylenums{`*цифры*`}`,

которая действует как в тексте, так и в формулах. В ее аргументе должны находиться только цифры и знаки препинания, результат печати других символов и букв непредсказуем. Цифры старого стиля имеются во всех шрифтах: `\oldstylenums{…}` ⇝ 1234567890, *1234567890*, 1234567890, *1234567890*, 1234567890, *1234567890*, **1234567890**….

2.8. Вывод текста программы

Для вывода небольших фрагментов рукописи, содержащих команды и служебные символы, определены команды `\verb`•*текст*• и `\verb*`•*текст*•, имеющие особый синтаксис. На месте кружков, служащих для выделения аргумента команд, можно использовать любой символ, отсутствующий в *тексте*, например `\verb+$x=y$+`. Аргумент (*текст*) должен умещаться в пределах одной строки. Он печатается как единое целое моноширинным шрифтом, при этом `\verb*` отличается тем, что использует символ ␣ для печати пробелов.

Команды `\verb` использованы нами для верстки примеров в абзацах: `\verb|$\alpha_i^2$|` ⇝ `$\alpha_i^2$` ⇝ α_i^2.

Для вывода нескольких строк или даже страниц рукописи имеются окружения `verbatim` и `verbatim*`, воспроизводящие текст именно так, как он набран, включая пробелы и разбиение на строки. С их помощью в громоздких примерах книги печатался текст рукописи, как показано на с.51. Как и команды, окружения различаются тем, что `verbatim*` печатает пробелы символом ␣. Окружения, определенные ядром LaTeX, ограничивают печатаемый текст пределами одной страницы, но загрузка стандартного пакета `verbatim` [20] снимает это ограничение. Вдобавок к этому пакет вводит команды `\verbatiminput{`*файл*`}` и `\verbatiminput*{`*файл*`}`, печатающие текст, загружаемый из *файла*, так, как это делают окружения `verbatim` и `verbatim*`. Для обширных комментариев он определяет окружение `comment`.

Важно отметить еще одно существенное ограничение: команды `\verb` и окружения `verbatim` не могут находиться в аргументах и параметрах других команд и окружений.

Глава 3

Рисунки и таблицы

Рисунки и таблицы состоят из непосредственно иллюстрации или таблицы и подписи к ним. Каждая составляющая обрабатывается отдельно по своим правилам.

Положение рисунков и таблиц в документе редко соответствует порядку их следования в рукописи. Как правило, они помещаются вверху или внизу страницы, или выносятся на специальные страницы, содержащие только рисунки и таблицы. Их положение меняется, в зависимости от заполненности страниц, они как бы «плавают» в документе и потому называются *плавающими объектами.*

В данном разделе мы рассмотрим ресурсы, которыми располагает LaTeX для работы с иллюстрациями, затем опишем средства, позволяющие формировать таблицы, а в конце обсудим компоновку плавающих объектов и средства управления их положением в документе.

3.1. Иллюстрации

Иллюстрации загружаются из графических файлов, обычно находящихся в рабочей папке вместе с документом. Если иллюстраций много, для них можно создать отдельную папку.

Компилятор считывает из файла информацию о размерах картинки, если необходимо, преобразует ее и вычисляет новый размер. Если же нужной информации в файле нет, он выдаст сообщение об ошибке. Размер задают координаты нижнего левого и правого верхнего

угла прямоугольника, в который вписана картинка. Как правило, они хранятся в специальном разделе файла, называемом BoundingBox.

В pdf-документ иллюстрация вставляется непосредственно при компиляции. В остальных случаях она замещается пустым боксом вычисленного размера, адресом файла, содержащего иллюстрацию, и описанием последовательности преобразований графических данных. Тогда для встраивания графики в документ дополнительно используется постпроцессор, например программа dvips (см. рис. 1.3).

Форматы графики зависят от формата генерируемого документа. В PostScript-документы можно вставить рисунки в формате PostScript (файлы с расширением •.ps и •.eps), а также в форматах растровых изображений Windows Bitmap и PCExchange (файлы с расширением •.bmp, •.pcx). Документы с расширением •.pdf могут содержать графику в форматах Portable Document Format (•.pdf), Joint Photographic Experts Group (•.jpg и •.jpeg), Portable Network Graphics (•.png) и Lossy/Lossless Coding of Bi-Level Images (•.jbig2 и •.jb2).

Универсальным форматом, поддерживающим как векторную, так растровую графику является PostScript и его подмножество pdf. С созданием и обработкой рисунков в данном формате в ОС Windows обычно возникают проблемы. Для из разрешения можно воспользоваться универсальным методом, описанным в прил. А.6. Имеющийся PostScript рисунок можно преобразовать в pdf-формат программой epstopdf, а пакет epstopdf позволяет это делать «на лету», непосредственно во время компиляции (см. разд. 7.9).

3.1.1. Загрузка иллюстраций

Для работы с графикой необходимо загрузить пакет graphicx из коллекции latex-graphics [21], определяющий команды

```
\includegraphics[параметры]{имя файла},
\includegraphics*[параметры]{имя файла}.
```

Обе они вставляют иллюстрации и различаются лишь тем, что первая выводит содержимое графического файла целиком, а вторая — только часть, находящуюся в пределах BoundingBox. Нижняя граница иллюстрации, точнее нижняя граница ее BoundingBox, выравнивается с базисной линией строки, в которую она встраивается.

В аргументе команд указывается имя файла, находящегося в рабочей папке, или путь к нему. Путь может быть абсолютным и от-

носительным. Последний указывается относительно рабочей папки, например `figures/file`, если рисунки собраны в папке `figures`. С помощью команды `\graphicspath{`*список папок*`}` можно задать список папок, в которых компилятор будет искать рисунки. Для этих рисунков указывается только имя файла. *Папки списка* заключаются в фигурные скобки: `\graphicspath{{/eps}{/pdf}}`.

Расширение имени файла лучше опустить, тогда компилятор сам будет искать файл допустимого графического формата. При верстке PostScript-документов поиск идет среди файлов с расширениями •.bmp, •.pcx, •.eps или •.ps, а для pdf-документов — •.pdf, •.jpg, •.jpeg, •.png и •.jbig2 или •.jb2. Если в папке имеются файлы picture.eps и picture.pdf, команда `\includegraphics{picture}` использует первый в PostScript, а второй — в pdf-документе. Использование же имени `picture.eps` при генерации pdf-документа даст ошибку, так как форматы иллюстрации и документа не будут соответствовать друг другу.

Параметры команды `\includegraphics` используются для обработки иллюстрации. Они перечисляются через запятую. Очередность их следования определяет последовательность преобразований картинки. Полный перечень параметров приведен в [3], перечислим лишь те, что позволяют вращать и масштабировать иллюстрации, а также корректировать их границы.

scale — масштабирование относительно размеров BoundingBox, Например, `scale=0.7` означает уменьшение до 70 % натурального размера иллюстрации.

width — масштабирование до заданной ширины, например до 75 % длины строки: `width=.75\linewidth`.

height — масштабирование до заданной высоты, например до 40 % высоты страницы: `height=.4\textheight`.

angle — поворот на угол, заданный в градусах, например `angle=7`. Положительные и отрицательные значения соответствуют вращению по часовой стрелке и против нее.

origin — положение центра вращения, заданное координатами, привязанными к BoundingBox и/или к базисной линии строки. Картинка вращается относительно своего левого нижнего угла, а при значении `origin=c` — относительно своего центра. Первая буква — горизонтальная координата, которая может принимать значения `l`, `r` и `c`, что соответствует левой, правой границе и середине ширины BoundingBox. Вторая буква — вертикальная координата.

Ее значения `t` и `b` привязывают центр вращения к верхней и нижней границе BoundingBox, `c` — к середине его высоты, а `B` — к положению базисной линии.

trim — корректировка BoundingBox. Указываются величины смещений левой, нижней, правой и верхней границ картинки. Положительные значения соответствуют обрезанию, а отрицательные — увеличению размеров. Например, `trim= 1mm 0 0 -4mm` означает обрезание левой и увеличение верхней границ картинки. Если числа указаны без размерности, по умолчанию используется большой пункт bp, равный 0.3528 мм.

clip — обрезание картинки по границам BoundingBox. Этот параметр автоматически используется командой `\includegraphics*`.

Когда параметры height и width используются по отдельности, размеры иллюстрации изменяются пропорционально, а при совместном использовании они подгоняются до заданных величин без сохранения пропорциональности.

Параметры trim и clip позволяют внести коррекцию, если графический файл сделан неправильно, или вырезать из него часть, которую необходимо вывести в документ. При этом удобно поместить иллюстрацию в рамку, которая позволит увидеть как размеры BoundingBox, так и положение картинки в отведенном для нее прямоугольнике. Рисует рамку команда `\fbox{`*текст*`}`. Между рамкой и заключенным в ней объектом делается зазор, устанавливаемый длиной `\fboxsep`. В нашем случае его можно сократить до минимума, достаточного, чтобы рамка не слилась с картинкой, поэтому, перед тем как рисовать рамку, нужно уменьшить данную длину (см. изменение длин на с. 37).

Разберем пример корректировки границ иллюстраций, показанных на рис. 3.1. Рисунок создает обсуждаемое далее окружение figure, в которое помещаются иллюстрации и подпись, печатаемая командой `\caption`. Слева на рисунке приведена исходная картинка, состоящая из стрелок, обозначающих углы BoundingBox, и цветного прямоугольника. Для наглядности она помещена в рамку, отделенную от нее зазором шириной 0.5pt. Та же иллюстрация с размерами, отредактированными с помощью параметра trim, выведена еще в трех видах. Сначала она показана с необрезанными полями. Хотя ее новые размеры соответствуют пространству внутри рамки, стрелки также печатаются, и чтобы они не наложились на соседние картинки, между ними вставлены пробелы `\quad`. Поля двух других картинок обрезаны параметром clip и

командой \includegraphics*. Третья иллюстрация помещена в рамку, уменьшена с коэффициентом 0.575 и повернута на 90°. Последняя картинка выведена без рамки. С помощью параметров width и height, установленных в одинаковый размер, она преобразована в квадрат.

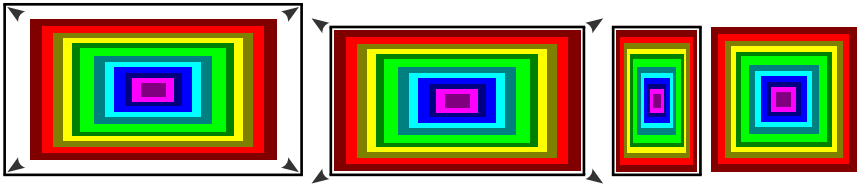


Рис. 3.1. Обработка иллюстраций с некорректными границами:

```
\begin{figure}[h!]
   \setlength\fboxsep{0.5pt}
   \fbox{\includegraphics{picture}}
   \quad
   \fbox{\includegraphics[trim= 2.9mm 1.5mm 2.9mm 1.5mm]{picture}}
   \quad
   \fbox{\includegraphics[trim= 2.9mm 1.5mm 2.9mm 1.5mm,%
                          scale=.575, angle=90, clip]{picture}}
   \includegraphics*[trim= trim= 2.9mm 1.5mm 2.9mm 1.5mm,%
                          width=5.5em, height=5.5em]{picture}
   \caption{Обработка иллюстраций с некорректными границами.}
   \label{f:picture}
\end{figure}
```

3.2. Верстка таблиц

LaTeX использует два метода создания таблиц. В первом задается шаблон строки с ячейками фиксированной ширины, по которому затем формируются все строки. Таким образом верстается таблица с фиксированной шириной колонок. Ее строки между собой не связаны, поэтому ее части могут располагаться на разных страницах. Данный метод реализован в окружении tabbing, описанном в книгах [2–4, 8].

Во втором методе шаблон строки задает только метод выравнивания и разделения ячеек, размеры которых определяют размеры помещенного в них текста. Ширину колонки определяет ее самая широкая ячейка. В большинстве случаев ширина колонок изменяется по мере заполнения таблицы. Сверстанная таблица становится боксом, встраиваемым в текстовую строку по общим правилам.

Второй метод, обладающий существенно большей гибкостью и универсальностью, реализован с помощью окружений array и tabular, имеющих одинаковый синтаксис. Первое используется в формулах, а второе в тексте. Этот метод используется чаще всего, поэтому рассмотрим его подробно, но сначала сформулируем основные правила верстки таблиц:

```
begin{tabular}{|l||c|r|}
                        \hline
\tiny 2 & 3       \\    \cline{1-2}
10      & 100     &  1000     \\
                        \cline{1-1}\cline{3-3}
$y_i^2$ & \tiger & \fbox{a} \\
                        \hline\hline
        & \bf ab & cd        \\ \hline
\end{tabular}
```

2	3	
10	100	1000
y_i^2		a
	ab	cd

Колонки с левым, правым выравниванием и центрированием обозначаются идентификаторами l, r и c. Список идентификаторов помещается в аргумент окружения tabular, который определяет число колонок, метод их выравнивания и порядок следования. В приведенном примере формируется таблица с тремя колонками, выровненными слева, по центру и справа.

Ячейки разделяются символом &, а строки командой \\. В конце строки пустые ячейки можно опустить, как это сделано в третьей строке примера, поставив команду \\ после последней заполненной ячейки. В начале строки пустые ячейки указываются всегда, так как это необходимо для правильного подсчета колонок. Количество строк в таблице не регламентируется, их можно свободно добавлять и удалять.

Ячейки должны содержать часть текстовой строки, в которой могут находиться строчные формулы, боксы, иллюстрации, как, например, в третьей строке таблицы, а также министраницы и даже другие таблицы. Каждая ячейка составляет группу, поэтому изменения шрифтов и других параметров, сделанные в одной из них, в другие не передаются. Это демонстрируют ячейки первой и последней строк таблицы.

Строки и колонки можно отчеркивать. Для этого в аргументе окружения tabular используется символ-разделитель |, рисующий вертикальную линию сверху донизу таблицы, а команда \hline проводит горизонтальную линию на всю ее ширину. Команды \hline должны стоять в начале строки. Количество отчеркивающих линий соответствует числу использованных команд \hline или разделителей |. Линии не накладываются друг на друга, а рисуются с небольшим зазором.

Если пустые ячейки в конце строки опущены, как в нашем примере, вертикальные линии справа от них тоже опускаются. Команда

`\cline{`*диапазон колонок*`}`

отчеркивает часть колонок. Она должна стоять после перевода строки. В ее аргументе указываются номера колонок, разделенные дефисом, как `\cline{1-2}` в примере, а если отчеркивается одна колонка, ее номер указывается дважды, см. `\cline{1-1}` и `\cline{3-3}` в примере. Чтобы отчеркнуть несколько колонок, нужно использовать несколько команд `\cline`, перечисляемых в произвольном порядке. Линии `\cline`, отчеркивающие одну и ту же колонку, накладываются друг на друга, а также и на линию `\hline`, поэтому команды `\hline` и `\cline` не стоит использовать вместе.

3.2.1. Выравнивание и разделение колонок

Рассмотрим синтаксис окружения tabular в деталях:

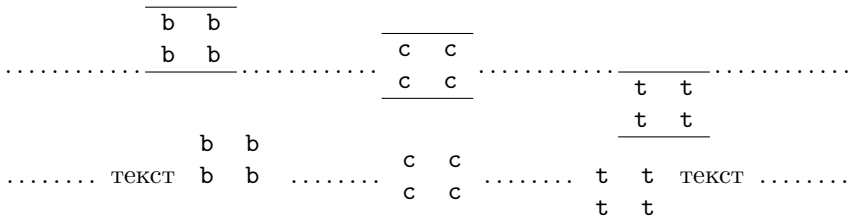
`\begin{tabular}[`*привязка*`]{`*шаблон строки*`}`

```
ячейка 1  &  ячейка 2  &  ... \\
```

Параметр *привязка*, определяющий встраивание таблицы в строку текста, может принимать следующие значения:

- b — встраивание последней строки таблицы в строку текста;
- **c** — центрирование таблицы относительно базисной линии;
- t — встраивание первой строки таблицы в строку текста.

По умолчанию бокс с таблицей центрируется в строке. Если таблица отчеркнута сверху или снизу, параметры b или t выравнивают отчеркивающие линии с базисной линией. Это демонстрируют следующие примеры:



В аргументе окружения tabular приводится список идентификаторов колонок и разделителей, перечисленных в табл. 3.1. Пробелы, поставленные между ними, не влияют на форматирование таблицы.

Таблица 3.1

Идентификаторы и разделители колонок окружения tabular

Идентификаторы	Выравнивание	Разделители	Вставки
c	по центру	|	вертикальная линия
l	левое	@{текст}	текст
r	правое		
p{ширина}	с обеих сторон по заданной ширине колонки		

По умолчанию колонки разделяются небольшим пустым полем, в котором разделитель `|` рисует вертикальную линию. Разделитель `@{текст}` заменяет пустое поле *текстом*. Поместив в него пробел, можно регулировать ширину поля, разделяющего колонки, как перед последним столбцом в данном примере:

```
\begin{tabular}{|l l | r @{*} l @{\hspace{3em}} l|} \hline
   1 & 2 & 3 & 4 & 5 \\
   6 & 7 & 8 & 9 & 0 \\ \hline
\end{tabular}
```

1	2	3*4	5
6	7	8*9	0

Уже говорилось, что ячейки колонок `c, l` и `r` содержат лишь одну строку текста. Ячейки колонок `p{ширина}` имеют фиксированную ширину, и потому могут содержать несколько строк. Помещенный в них текст автоматически нарезается на строки заданной длины и выравнивается с обеих сторон. Для разрыва строк можно пользоваться командами `\linebreak[•]` и `\newline`. В строку таблицы встраивается первая строка ячеек `p{•}`:

```
\begin{tabular}{lp{8em}}
\hline
  1 & Короткая строка.\\
  2 & Более длинная строка.\\
  3 & Длинный текст разбивается...\\
\hline
\end{tabular}
```

1	Короткая строка.
2	Более длинная строка.
3	Длинный текст разбивается на несколько строк.

3.2.2. Выравнивание ячеек

Колонки `c, l` и `r` выравниваются пружинами `\hfil`, неявно стоящими в каждой ячейке. В колонках `l` и `r` они, соответственно, стоят справа и слева, а центрированную колонку сжимают с обеих сторон.

Используя дополнительные пружины, можно регулировать выравнивание отдельных ячеек, не затрагивая при этом другие:

```
\begin{tabular}{|l|c|r|}          \hline
       1 &        2 & 3               \\
    1111 &     2222 & 3333            \\
\hfil  1 & \hfil  2 & 3 \hfil\mbox{}  \\
\hfill 1 & \hfill 2 & 3 \hfill\mbox{} \\
                                  \hline
\end{tabular}
```

1	2	3
1111	2222	3333
1	2	3
1	2	3

Разберем приведенный выше пример. Первая и вторая строка таблицы иллюстрируют методы выравнивания колонок. В ячейки третьей строки мы поставили пружины `\hfil`, при этом в крайних колонках, выравниваемых слева и справа, цифры, оказавшись между одинаковыми пружинами, сместились в центр. В центрированной колонке двойка смещена вправо на 2/3 пустого поля, так как слева она зажата двумя одинаковыми пружинами, а справа — одной. Обратите внимание, что только левая граница колонки служит упором пружин, справа же он отсутствует, поэтому в последней колонке в качестве упора использован пустой бокс. В ячейки четвертой строки вставлены жесткие пружины `\hfill`, которые смяли все «неявные» пружины и отжали цифры к границам колонок.

3.2.3. Объединение колонок

Ячейки можно объединять как в строках, так и в колонках. В строке ячейки объединяет команда

`\multicolumn{`*n*`}{`*описание ячейки*`}{`*текст*`}`

Она имеет три аргумента. В первом указывается число объединяемых ячеек, во втором — разделители, окружающие объединенную ячейку, и метод ее выравнивания, задаваемый одним из идентификаторов. В третий аргумент помещается текст ячейки. Используем команду `\multicolumn`, чтобы сформировать заголовок таблицы:

Физические	Единицы		Соотношение
величины	СИ	СГС	единиц
Длина	м	см	1 м = 100 см
Масса	кг	г	1 кг = 1000 г
Энергия	Дж	эрг	1 Дж = 10^7 эрг

```
\begin{tabular}{ lccr @{ } l}                    \toprule
  Физические & \multicolumn{2}{c}{Единицы} &
         \multicolumn{2}{c}{Соотношение}   \\ \cmidrule{2-3}
  величины & СИ & СГС &
              \multicolumn{2}{c}{единиц}   \\ \midrule
  Длина   & м  & см   & 1 м  = & 100 см     \\
  Масса   & кг & г    & 1 кг = & 1000 г     \\
  Энергия & Дж & эрг  & 1 Дж = & $10^7$ эрг \\ \bottomrule
\end{tabular}
```

В первой строке таблицы заголовком «Единицы» объединены вторая и третья колонки, содержащие обозначения двух систем единиц физических величин. Заголовок «Соотношение единиц» сделан путем объединения четвертой и пятой колонок в первой и второй строке. Между четвертой и пятой колонками с помощью разделителя `@{ }` вставлен пробел, к которому они прижимаются справа и слева. Заголовок таблицы, сформированный таким образом, выглядит рыхлым. Далее мы покажем, как улучшить его вид.

В соответствии с правилами оформления таблиц в журнальных публикациях в нашем примере отсутствуют вертикальные отчеркивающие линии. Для отделения заголовка и отчеркивания таблицы использованы команды `\midrule[длина]`, `\toprule[длина]` и `\bottomrule[длина]`, вводимые пакетом booktabs [22]. Они заменяют стандартную команду `\hline`, во многих случаях давая лучший результат. Их параметр позволяет регулировать толщину линий.

Для отчеркивания ячеек данный пакет определяет команду

`\cmidrule[`*толщина*`](`*ширина*`){`*диапазон колонок*`}`,

позволяющую регулировать не только толщину линии, но и ее ширину. Ее второй параметр, заключенный в нестандартные круглые скобки, задает обрезание отчеркивающей линии слева и справа. Ширину обрезания можно задать как `r{`*длина*`}`, `l{`*длина*`}`, `lr{`*длина*`}` или просто `r`, `l`, `lr`. В последнем случае используется длина `\cmidrulekern`, по умолчанию равная 0.5em.

3.2.4. Объединение строк

Непосредственно в ядре LaTeX отсутствуют средства объединения ячеек в колонках. Такую возможность предоставляет пакет multirow, который вводит с этой целью команду `\multirow`. Объединенная ячейка может содержать несколько строк текста. Покажем это на примере:

```
\begin{tabular}{|l|l|p{0.5em}|l|}   \hline
   1 & 2 & 3 & 4                  \\ \hline
   2 & \multirow{3}*{a}
         & \multirow{3}={a \\ b}
             &                    \\
   3 &   &   &                    \\
   4 &   &   &
   \multirow{-3}{0.5em}{a \\[-.75ex] b \\[-1ex] c\\[-.75ex] d}
                                  \\ \hline
\end{tabular}
```

1	2	3	4
2 3 4	a	a b	a b c d

В первом столбце и строке таблицы мы пронумеровали ячейки, а в остальных создали ячейки, объединяющие три строки каждой колонки. Текст ячеек второго, третьего и четвертого столбца состоит, соответственно, из одной, двух и четырех строк. Скорректировав межстрочный интервал текста в последней ячейке, мы вписали ее в высоту колонки. Для разделения строк объединенной ячейки можно пользоваться командами `\\`, `\newline` и `\linebreak`.

Разберем теперь синтаксис и действие команды

`\multirow[`*привязка*`]{`*n*`}[`*распорки*`]{`*ширина*`}[`*смещение*`]{`*текст*`}`

Ее первый параметр определяет метод вертикального выравнивания объединенной ячейки, который задается идентификаторами:

- **c** центрирование (значение по умолчанию);
- b выравнивание последней строки текста ячейки с последней объединяемой строкой колонки;
- t встраивание первой строки текста ячейки с первой объединяемой строкой колонки.

Редко используемый параметр *распорки* настраивает высоту объединяемых строк по сложному алгоритму с замысловатой логикой, описанной в документации пакета multirow [23].

Параметр *смещение*, имеющий размерность длины, задает высоту, на которую можно сместить текст ячейки. При положительной длине текст смещается вверх, а при отрицательной вниз.

В первом аргументе команды `\mutirow` указывается количество объединяемых ячеек, причем задающее его число, как показано в примере, может быть положительно или отрицательно. При положительном значении команда `\mutirow` должна стоять в первой объединяемой ячейке, а при отрицательном — в последней. В любом случае остальные ячейки должны быть пусты. Текст объединенной ячейки печатается вместе со строкой таблицы, в которой находится команда `\mutirow`. Порядок его

вывода может влиять на результат верстки. Например, если колонка заливается фоном, а команда `\mutirow` находится в первой ячейке, фон закроет текст, если же она находится в последней ячейке, текст будет напечатан поверх фона.

Второй аргумент регулирует ширину объединенной ячейки. Указав необходимую длину, ее можно задать явно. Специальное значение `=` позволяет наследовать ширину колонки `p{•}`, как это сделано в примере для третьей колонки. Еще одно специальное значение `*` создает ячейку, наследующую ширину колонок `l`, `r` или `c`. Такая ячейка на строки не разбивается. Все три варианта настройки ширины ячейки продемонстрированы в нашем примере.

Декларация `\multirowsetup` хранит код, выполняемый перед верстской ячейки `\multirow`, который по умолчанию состоит из декларации `\raggedright`, обеспечивающей левое выравнивание текста. Изменив данную команду (см. разд. 7.1), можно настроить параметры верстки всех объединяемых ячеек, например заменить левое выравнивание центрированием: `\renewcommand\multirowsetup{\centering}` . В отдельных ячейках это легко сделать, поставив нужные декларации перед их текстом.

3.2.5. Объединение строк и колонок

Если понадобится ячейка, объединяющая как строки, так и колонки, используйте для ее построения команду `\multirow` в аргументе команды `\multicolumn`. Например,

```
\multicolumn{2}{c}{\multirow{3}{*}{Multi-multi}}.
```

Обратная комбинация работать не будет.

Воспользуемся такой возможностью, чтобы улучшить заголовок сделанной ранее таблицы физических величин:

Физические величины	Единицы		Соотношение единиц
	СИ	СГС	

```
\begin{tabular}{ lccr @{ } l}   \toprule
  \multirow{2}{6em}{\centering\bigstrut[t] Физические величины} &
    \multicolumn{2}{c}{Единицы} &
    \multicolumn{2}{c}{
       \multirow{2}{7em}{\centering\bigstrut[t] Соотношение единиц}}
              \\ \cmidrule{2-3}
  & СИ & СГС &  \\ \midrule
```

В первой колонке объединены только строки, во второй и третьей колонках — ячейки первой строки, а в четвертой и пятой объединяются и строки, и колонки. Обратите внимание на метод выравнивания первого и последнего заголовков. Их центрируют декларации `\centering`, поставленные в объединенные ячейки, а команды `\bigstrut[t]` делают небольшую верхнюю отбивку. Команды `\bigstrut[`*смещение*`]`, определяемые пакетом bigstrut [23], вставляют распорку, увеличивающую высоту и глубину строки. Значения параметра `b` и `t` регулируют смещение, изменяя либо только глубину, либо только высоту. Величину смещения задает длина `\bigstrutjot`, по умолчанию равная 2pt.

На этом закончим обсуждение основных ресурсов верстки таблиц. Дополнительные возможности описаны в гл. 10.

3.3. Плавающие объекты

Наблюдательный читатель возможно обратил внимание, что в данной книге рисунки и таблицы находятся рядом со ссылкой на них, что соответствует правилам российского книгоиздания. Однако такое расположение не типично для LaTeX.

Рисунки или таблицы обычно имеют довольно большие размеры и потому их помещают в «плавающие» объекты, положение которых в рукописи и документе может отличаться. Компилятор пытается оптимально распределить плавающие объекты, размещая их обычно вверху страниц поблизости от того места, которое они занимают в рукописи, так как там они требуют минимум места.[1] Если же их количество велико, а объем текста мал, они сводятся все вместе в конец документа. Использование в рукописи статьи средств, управляющих размещением плавающих объектов, не приветствуется. При необходимости их расставит по местам технический редактор.

На примере рис. 3.2 покажем автоматическое размещение плавающего объекта по правилам LaTeX. В тексте рукописи он находится *непосредственно после* этого абзаца.

Из примера видно, что помимо самой иллюстрации плавающий объект может содержать подпись, метку и любой дополнительный текст, который в примере представлен кодом, генерирующим рисунок.

[1] Рисунки и таблицы отбиваются от окружающего текста вертикальными пробелами, которые в начале и конце страницы игнорируются, поэтому они прижимаются к краю страницы, экономя место для текста.

Рис. 3.2. Стандартная компоновка рисунка:

```
\begin{figure}
  \centering
  \includegraphics{tiger}
  \caption{Стандартная компоновка рисунка:}
  \label{f:tiger}
\end{figure}
```

Плавающие объекты создают окружения, предназначенные для размещения таблиц и рисунков:

`\begin{table}`[положение],	`\begin{table*}`[положение],
`\begin{figure}`[положение],	`\begin{figure*}`[положение]

При многоколоночном наборе рисунки и таблицы, помещенные в окружения figure* или table*, занимают всю ширину страницы, а в figure или table — ширину колонки.

Положение плавающего объекта, который обычно выводится вверху страницы, настраивает параметр окружений. В нем можно задать один или несколько идентификаторов, например `\begin{figure}[tbh]`, опробовав которые, компилятор выберет оптимальный. Идентификаторы имеют следующие значения:

- **t** — поместить вверху колонки или страницы;
- b — поместить внизу колонки или страницы;
- h — разместить в соответствии с порядком следования в тексте;
- p — вынести на специальную страницу, содержащую плавающие объекты.

Распределение плавающих объектов по страницам сильно влияет на качество верстки документа в целом, поэтому их размещение не всегда следует заданному параметру. Модификатор `!` усиливает его действие. Например, таблица, помещенная в окружение `\begin{table}[h!]`, должна появиться среди текста, окружающего ее в рукописи. В большинстве случаев модификатор `!` позволяет добиться желаемого результата, но если сверстанная страница окажется слиш-

ком пустой или переполненной, параметр все же может игнорироваться. Дополнительные средства, управляющие положением плавающих объектов обсуждаются в разд. 8.6.

Рисунок или таблицу можно подписать с помощью команды

`\caption[`*подпись в списках*`]{`*подпись в тексте*`}`.

Ее использование вне окружений figure или tabular вызовет ошибку компиляции, так как она определена только внутри них. Аргумент команды содержит подпись, печатаемую в тексте, а в параметре указывают ее вариант для в списков рисунков или таблиц. В документах, написанных по-русски, в конце подписи точка не ставится, а в англоязычных ее, как правило, ставят.

Параметр позволяет убрать из подписи элементы, которые не должны отражаться в списках. Например, табл. 5.1 имеет примечание, а ее подпись содержит команду `\footnotemark`, генерирующую номер примечания. В то же время в списке таблиц ссылки на примечание нет, так как в варианте подписи, помещенном в параметр команды `\caption`, команда `\footnotemark` отсутствует.

С командами `\caption` ассоциированы счетчики figure и table. Именно подпись является объектом ссылки на рисунок или таблицу, поэтому непосредственно после команды `\caption` следует поставить метку. Рекомендуем снабдить ее префиксом, например `\label{f:•}` или `\label{t:•}`. Это облегчит поиск нужной метки при создании ссылок. Например, в программе TEXstudio список меток сортируется и одинаковые префиксы группируются вместе.

Плавающий объект компонуется как обычный текст, в котором рисунки и таблицы представляют собой боксы, находящиеся в текстовой строке, а подпись является отдельным абзацем. Положение подписи относительно рисунка (или таблицы) задает порядок следования команд `\caption` и `\includegraphics` (или окружения tabular). Если команда `\caption` стоит впереди, подпись выводится сверху, а если позади, то снизу, как показано ранее.

Чтобы вывести подпись сбоку от рисунка, нужно использовать министраницы. Разберем конструкцию рис. 3.3, обеспечивающую такую компоновку. Ее основу составляют две министраницы, стоящие в одной строке. Чтобы они в ней уместились, их суммарная ширина сделана меньше длины строки. В первую министраницу помещена подпись, а во второй находится картинка. Для их выравнивания применена декларация `\centering`, а между министраницами вставлена пружина

Рис. 3.3. Компоновка рисунка и подписи, расположенной слева:

```
\begin{figure}
 \centering
  \begin{minipage}{.45\linewidth}
     \caption[Компоновка рисунка и подписи, расположенной слева]%
             {Компоновка рисунка и подписи, расположенной слева:}
     \label{f:TigerAtTheRight}
  \end{minipage}
\hfil
  \begin{minipage}{.25\linewidthh}
     \includegraphics[width=\linewidth]{tiger}
  \end{minipage}
\end{figure}
```

Рис. 3.4. Компоновка рисунка, состоящего из двух картинок и подписи, расположенной справа от них:

```
\begin{figure}
  \begin{minipage}{.52\linewidth}
     \reflectbox{\includegraphics[width=.48\linewidth]{tiger}}
     \hfill      \includegraphics[width=.48\linewidth]{tiger}
  \end{minipage}
\hfill
  \begin{minipage}{.4\linewidth}
     \caption[Компоновка рисунка, состоящего из двух картинок и
                подписи, расположенной справа от них]%
             {Компоновка рисунка, состоящего из двух картинок и
                подписи, расположенной справа от них:}
     \label{f:Mirror}
  \end{minipage}
\end{figure}
```

`\hfil`. Вместе они создают одинаковую отбивку рисунка и подписи относительно границ страницы и друг друга.

Ширина левой министраницы задает длину подписи, а правой — размер картинки. Длина строки министраницы равна ее ширине, поэтому ширина картинки, заданная как `width=\linewidth`, становится равной этому размеру. Изменяя ширину министраниц, удобно регулировать размеры рисунка и подписи.

По умолчанию обе министраницы центрируются относительно базисной линии строки, поэтому подпись автоматически центрируется по высоте рисунка.

Усложним компоновку, добавив еще одну картинку на рис. 3.4, и чтобы предать композиции законченность, отразим ее, воспользовавшись командой `\reflectbox{`*текст*`}`. Ее аргументом может быть строка *текста* (см. разд. 7.3).

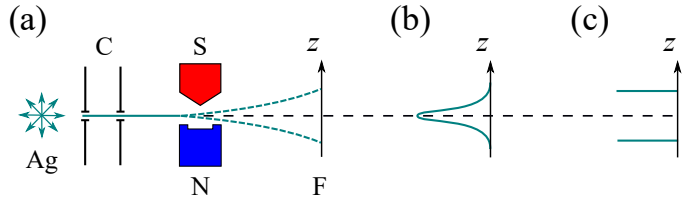
Чтобы уместить пару картинок, мы увеличили ширину министраницы, а чтобы тигры не щекотали друг друга усами, вставили между ними жесткую пружину `\hfill`, разводящую их по краям министраницы. Такая же пружина отжимает картинки и подпись к краям страницы.

Приведем без комментариев показанный на рис. 3.5 вариант удобной компоновки иллюстраций с помощью таблицы.

Рис. 3.5. Кувырок:

```
\begin{figure}[h!] \centering
   \begin{tabular}{ccccc}
      I & II & III & IV & V \\
      \includegraphics[scale=.2]{tiger}                       &
      \includegraphics[scale=.2, origin=c, angle=-90]{tiger}  &
      \includegraphics[scale=.2, origin=c, angle=-180]{tiger} &
      \includegraphics[scale=.2, origin=c, angle=-270]{tiger} &
      \reflectbox{\includegraphics[scale=.2]{tiger}}
   \end{tabular}
   \caption[Кувырок]{Кувырок:} \label{f:Tumble}
\end{figure}
```

Часто рисунки требуют пояснений, которые приводят после их первого упоминания в тексте или располагают в самих рисунках. Пояснения лучше вносить в подпись, но их также можно напечатать над ней, как показано на рис. 3.6, или сбоку от иллюстраций. Плавающие объекты допускают любую компоновку подписей, пояснений, иллюстраций и таблиц.



(a) — схема эксперимента: пучок атомов серебра, формируемый коллиматором C пролетает через градиентный магнит S-N и осаждается на экране F. (b) и (c) — распределение интенсивности пучка, ожидаемое, соответственно, в классическом и квантовом случаях. Пунктирная горизонтальная линия — траектория неотклоненного пучка в нулевом магнитном поле.

Рис. 3.6. Схема эксперимента Штерна-Герлаха:

```
\begin{figure}[h!]
   \centerline{\includegraphics{figures/SternGerlach}}

\smallskip\small
(a) --- схема эксперимента: ...

\caption[Схема эксперимента Штерна-Герлаха]%
        {Схема эксперимент Штерна-Герлаха:}\label{f:Stern-Gerlach}
\end{figure}
```

В данном примере, чтобы не нарушать выравнивание текста пояснений, иллюстрацию центрирует команда `\centerline`. Пояснение представляет собой обычный абзац без отступа, подавляемого в плавающих объектах. Команда `\smallskip` отбивает иллюстрацию снизу, а декларация `\small` выравнивает размер шрифта пояснения и подписи.

Глава 4

Верстка формул

Высокое качество верстки формул — одно из главных достоинств TEX. Алгоритмы обработки текста и математических выражений сильно различаются. Различаются также наборы шрифтов и команд, которыми оперирует компилятор, поэтому математические символы нельзя использовать в тексте, а русские буквы — в формулах. Большинство команд, используемых в верстке текста, неприменимо в формулах, и наоборот.

Основными элементами формул являются переменные, функции, операторы, скобки, знаки различных соотношений, индексы, корни, дроби и т. д. Каждый элемент имеет свое шрифтовое оформление, размер и отбивку, которые очень хорошо сбалансированы с окружающими элементами. Символы и основные конструкции задаются командами, а сложные конструкции типа матриц, систем уравнений и т. п.— окружениями. Пробелы также регулируются командами, а пробелы рукописи полностью игнорируются. Как правило, в формулах имеются команды, аргументы которых содержат другие команды со своими аргументами, и «уровень вложенности» таких команд может быть довольно велик. Например, на «третьем уровне» находятся индексы переменной, стоящей в числителе или знаменателе дроби, являющейся подкоренным выражением. Поскольку аргументы заключаются в фигурные скобки, возникает много вложенных скобок, а потеря хотя бы одной из них ведет к ошибке компиляции. Поэтому рекомендуем по возможности избегать лишних скобок и, используя алгоритм автоматического определения аргумента команд, набирать `x_i^2` вместо `x_{i}^{2}`, или

`\sqrt2` вместо `\sqrt{2}`. Это намного упростит текст рукописи и уменьшит вероятность ошибок. Набирая скобки, придерживайтесь золотого правила: вводите сразу пару — открывающую и закрывающую.

Набор доступных команд и окружений определяется списком загружаемых пакетов. К числу стандартных ресурсов принадлежит коллекция пакетов $\mathcal{AMS}$-LaTeX Американского математического общества. Описываемые далее средства верстки формул подразумевают использование пакетов amssymb, amsfonts [24] и amsmath [25] из этой коллекции. Первый из них автоматически загружает пакет amsfonts и существенно расширяет набор символов. Второй загружает пакеты amsbsy [26], amsopn [27], amstext [28] и определяет большое число дополнительных конструкций. Мы не будем разделять ресурсы стандартного LaTeX и $\mathcal{AMS}$-LaTeX,[1] поэтому подчеркнем еще раз, что преамбула документа должна содержать команду `\usepackage{amssymb,amsmath}`.

Простые формулы, находящиеся в текстовой строке, заключаются в символы `$...$`, например

`$\sqrt{x^2} = \pm x$` $\rightsquigarrow \sqrt{x^2} = \pm x$.

Будем называть их *строчными формулами.* Вместо символов `$` формулу можно заключить в команды `\(...\)` или в окружение math, но они менее удобны и потому практически не используются.

Громоздкие формулы выносятся из строки в отдельный абзац, и потому будут называться *вынесенными.* Приведем пример такой формулы:

$$\int_0^1 \frac{x^2 dx}{\sqrt{1-x^2}} = \frac{1}{2}\left(\arcsin x - x\sqrt{1-x^2}\right)\Big|_0^1 = \frac{\pi}{4}.$$

Вынесенные формулы бывают *нумерованные* и *ненумерованные.* Нумерованные формулы создаются с помощью окружений, обсуждаемых в разд. 4.4. Ненумерованные формулы заключаются либо в двойные символы `$$...$$`, либо между команд `\[...\]`. Конструкция `$$...$$` наследована из TeX, а `\[...\]` определена в LaTeX. Рекомендуем пользоваться последней из них, так как в некоторых случаях она обрабатывается более корректно. Например, при загрузке класса документа с параметром fleqn (см. с. 24) формулы `\[...\]` прижмутся к левому полю, тогда как формулы `$$...$$` останутся центрированными. Наряду с командами `\[...\]` можно пользоваться окружениями displaymath и equation*, что для громоздких формул более удобно.

[1] Окружения и команды, определяемые пакетами $\mathcal{AMS}$-LaTeX, помечены в предметном указателе.

Различные части формулы по мере необходимости поднимаются или опускаются относительное ее *базовой* линии. При этом формируются несколько строк с собственными базовыми линиями, например числитель и знаменатель дроби или разные строки математических блоков (см. разд. 4.3). Среди них можно выделить *основную* строку, открываемую в самом начале формулы, в которой находятся все ее части. Как и в тексте, к базовым линиям привязывается низ символов, не имеющих глубины, и знаки препинания. Наряду с базовыми имеются *осевые* линии, относительно которых некоторые части формул выравниваются по вертикали. Они проходят на уровне знаков минус или горизонтальной линии знаков плюс. В приведенном выше примере можно заметить, что разделительные линии дробей привязываются по вертикали к знакам минус основной строки формулы, а знаки равенства и интеграл центрируются относительно них.

При встраивании строчной формулы базовая линия ее основной строки совмещается с базовой линией строки текста.

Как строчные, так и вынесенные формулы являются либо частью фразы, либо самостоятельными «предложениями». В соответствии с окружающим контекстом они должны выделяться знаками пунктуации. В конце формулы, завершающей предложение, или самостоятельной формулы ставится точка, а отдельные части (строки) сложных формул разделяются либо запятыми, либо точками.

Чтобы не загромождать представленные далее примеры, в формулах опущены символы `$` или `\[` и `\]`. Приведены лишь команды, необходимые для понимания сути примеров.

4.1. Основные элементы и конструкции

4.1.1. Буквы, символы и векторы

В формулах определены латинские буквы, цифры и символы `( ) [ ] + − = ! ? | / < > * . , : ; ′ ` "`, вводимые непосредственно с клавиатуры. Большая часть математических символов, в том числе фигурные скобки и греческие буквы, задана командами. Чтобы не загромождать текст, в данном разделе приведены только сами символы, а полная информация о них вынесена в прил. В.

Буквы и символы могут быть жирной или обычной насыщенности. Жирные символы выводят команды `\boldsymbol{выражение}` и `\bm{выражение}`. Первую определяет пакет amsmath, а вторую —

стандартный пакет bm [29], который нужно загрузить для ее использования. Обе команды действуют одинаково, но «компактность» делает команду `\bm{•}` более удобной. Кроме того, ее код совершенствуется вместе с ядром LaTeX, поэтому лучше пользоваться именно ей.

Команды `\boldsymbol` и `\bm` делают жирным не отдельный символ, а все выражение. Результат их действия зависит от имеющихся шрифтов. Насыщенность символов не изменится, если в шрифтах отсутствуют их жирные варианты. Сравните

```
ABC\ne\mathbb{ABC}        ⇝   ABC ≠ 𝔸𝔹ℂ,
\bm{ABC\ne\mathbb{ABC}}   ⇝   𝑨𝑩𝑪 ≠ 𝔸𝔹ℂ.
```

Переменные

Для обозначений переменных LaTeX предоставляет широкий выбор начертаний латинских и греческих букв. Стандартным начертанием букв в формулах является курсив, а цифры и скобки печатаются прямым шрифтом. Шрифты других начертаний, называемые *математическими алфавитами*, представлены в табл. 4.1. Их буквы выводят команды, например,

```
\mathfrak{a...z, A...Z}        ⇝   𝔞 … 𝔷, 𝔄 … 𝔷,
\bm{\mathfrak{a...z, A...Z}}   ⇝   𝖆 … 𝖟, 𝕬 … 𝖅.
```

Таблица 4.1

Команды и символы математических алфавитов

Стандартный математический шрифт `\mathnormal{•}`:

$$A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z$$
$$a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l\,m\,n\,o\,p\,q\,r\,s\,t\,u\,v\,w\,x\,y\,z$$
$$0\,1\,2\,3\,4\,5\,6\,7\,8\,9$$

Жирный математический шрифт `\bm{•}`:

$$\boldsymbol{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$$
$$\boldsymbol{a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l\,m\,n\,o\,p\,q\,r\,s\,t\,u\,v\,w\,x\,y\,z}$$
$$\boldsymbol{0\,1\,2\,3\,4\,5\,6\,7\,8\,9}$$

Математический курсив `\mathit{•}`:

$$\mathit{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$$
$$\mathit{a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l\,m\,n\,o\,p\,q\,r\,s\,t\,u\,v\,w\,x\,y\,z}$$
$$\mathit{0\,1\,2\,3\,4\,5\,6\,7\,8\,9}$$

Продолжение табл. 4.1

Жирный математический курсив \bm{\mathit{•}}:

$\boldsymbol{\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}}$
$\boldsymbol{\mathit{abcdefghijklmnopqrstuvwxyz}}$
$\boldsymbol{\mathit{0123456789}}$

Прямой шрифт \mathrm{•} :

$\mathrm{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathrm{abcdefghijklmnopqrstuvwxyz}$
$\mathrm{0123456789}$

Жирный прямой шрифт \mathbf{•} :

$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathbf{abcdefghijklmnopqrstuvwxyz}$
$\mathbf{0123456789}$

Шрифт Typewriter \mathtt{•} :

$\mathtt{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathtt{abcdefghijklmnopqrstuvwxyz}$
$\mathtt{0123456789}$

Рубленый шрифт \mathsf{•} :

$\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
$\mathsf{abcdefghijklmnopqrstuvwxyz}$
$\mathsf{0123456789}$

Жирный рубленый шрифт \bm{\mathsf{•}}:

$\boldsymbol{\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}}$
$\boldsymbol{\mathsf{abcdefghijklmnopqrstuvwxyz}}$
$\boldsymbol{\mathsf{0123456789}}$

Каллиграфический шрифт \mathcal{•} :

$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

Жирный каллиграфический шрифт \bm{\mathcal{•}}:

$\boldsymbol{\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}}$

Окончание табл. 4.1

Рукописный шрифт `\mathscr{•}`:

$\mathscr{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$

Жирный рукописный шрифт `\bm{\mathscr{•}}`:

$\boldsymbol{\mathscr{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}}$

Готический шрифт `\mathfrak{•}`:

$\mathfrak{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$
$\mathfrak{a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l\,m\,n\,o\,p\,q\,r\,s\,t\,u\,v\,w\,x\,y\,z}$

Жирный готический шрифт `\bm{\mathfrak{•}}`:

$\boldsymbol{\mathfrak{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}}$
$\boldsymbol{\mathfrak{a\,b\,c\,d\,e\,f\,g\,h\,i\,j\,k\,l\,m\,n\,o\,p\,q\,r\,s\,t\,u\,v\,w\,x\,y\,z}}$

Ажурный шрифт `\mathbb{•}`:

$\mathbb{A\,B\,C\,D\,E\,F\,G\,H\,I\,J\,K\,L\,M\,N\,O\,P\,Q\,R\,S\,T\,U\,V\,W\,X\,Y\,Z}$

Все перечисленные алфавиты, кроме `\mathtt` и `\mathbb`, имеют жирные шрифты, активируемые командой `\bm`. Исключение составляет жирный прямой шрифт, имеющий собственную команду `\mathbf`.

Чтобы использовать рукописный шрифт вместе с каллиграфическим, загрузите пакет `eucal` с параметром `mathscr`. Загрузка без параметра заменит каллиграфический шрифт рукописным.

Символы, заданные командами, как правило, остаются неизменными при смене шрифта, `\mathfrak{x\le\alpha D}` $\rightsquigarrow \mathfrak{x} \le \alpha\mathfrak{D}$, тем не менее чтобы избежать конфуза, данную операцию стоит применять лишь к отдельным буквам, следя за корректностью ее выполнения. В шрифтах `\mathcal`, `\mahscr` и `\mathbb` есть только заглавные буквы, а на месте строчных букв и цифр стоят совершенно другие символы, поэтому их использование может привести к забавным результатам: `\mathbb{A_i}` $\rightsquigarrow \mathbb{A}_\beth$, или `\mathcal{x^2}` $\rightsquigarrow \S^{\in}$.

Переменные часто обозначаются греческими буквами, различные варианты которых собраны в табл. 4.2 и В.11. Каждую из них печатает собственная команда, например `\beta\Gamma` $\rightsquigarrow \beta\Gamma$, `\bm\beta` $\rightsquigarrow \boldsymbol{\beta}$. По умолчанию строчные буквы имеют курсивное начертание, а прописные буквы доступны как в прямом, так и курсивном начертании. Чтобы получить доступ к прямым строчным буквам загрузите пакет `upgreek`

[30] (см. с. 148). Он подключит дополнительные шрифты и определит команды с именами, начинающимися префиксом `\up...`, например `\upbeta` ⇝ $\upbeta$, `\bm\upbeta` ⇝ $\bm\upbeta$.

Таблица 4.2

Греческие буквы

$\Gamma\,\Delta\,\Theta\,\Lambda\,\Xi\,\Pi\,\Sigma\,\Upsilon\,\Phi\,\Psi\,\Omega \quad \mathit{\Gamma\,\Delta\,\Theta\,\Lambda\,\Xi\,\Pi\,\Sigma\,\Upsilon\,\Phi\,\Psi\,\Omega} \quad \mathit{F}$

$\alpha\,\beta\,\gamma\,\delta\,\varepsilon\,\epsilon\,\zeta\,\eta\,\theta\,\vartheta\,\iota\,\kappa\,\varkappa\,\lambda\,\mu\,\nu\,\xi\,o\,\pi\,\varpi\,\rho\,\varrho\,\varsigma\,\sigma\,\tau\,\upsilon\,\varphi\,\phi\,\chi\,\psi\,\omega$

$\upalpha\,\upbeta\,\upgamma\,\updelta\,\upvarepsilon\,\upepsilon\,\upzeta\,\upeta\,\uptheta\,\upvartheta\,\upiota\,\upkappa\,\upvarkappa\,\uplambda\,\upmu\,\upnu\,\upxi\,\upo\,\uppi\,\upvarpi\,\uprho\,\upvarrho\,\upvarsigma\,\upsigma\,\uptau\,\upupsilon\,\upvarphi\,\upphi\,\upchi\,\uppsi\,\upomega$

$\bm{\Gamma\,\Delta\,\Theta\,\Lambda\,\Xi\,\Pi\,\Sigma\,\Upsilon\,\Phi\,\Psi\,\Omega} \quad \bm{\mathit{\Gamma\,\Delta\,\Theta\,\Lambda\,\Xi\,\Pi\,\Sigma\,\Upsilon\,\Phi\,\Psi\,\Omega}}$

$\bm{\alpha\,\beta\,\gamma\,\delta\,\varepsilon\,\epsilon\,\zeta\,\eta\,\theta\,\vartheta\,\iota\,\kappa\,\varkappa\,\lambda\,\mu\,\nu\,\xi\,o\,\pi\,\varpi\,\rho\,\varrho\,\varsigma\,\sigma\,\tau\,\upsilon\,\varphi\,\phi\,\chi\,\psi\,\omega}$

$\bm{\upalpha\,\upbeta\,\upgamma\,\updelta\,\upvarepsilon\,\upepsilon\,\upzeta\,\upeta\,\uptheta\,\upvartheta\,\upiota\,\upkappa\,\upvarkappa\,\uplambda\,\upmu\,\upnu\,\upxi\,\upo\,\uppi\,\upvarpi\,\uprho\,\upvarrho\,\upvarsigma\,\upsigma\,\uptau\,\upupsilon\,\upvarphi\,\upphi\,\upchi\,\uppsi\,\upomega}$

Таблица 4.3

Математические акценты

•	Команды	•	Команды	•	Команды	•	Команды
$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\breve{a}$	`\breve{a}`	$\tilde{a}$	`\tilde{a}`
$\bar{a}$	`\bar{a}`	$\acute{a}$	`\acute{a}`	$\grave{a}$	`\grave{a}`	$\vec{a}$	`\vec{a}`
$\dot{a}$	`\dot{a}`	$\ddot{a}$	`\ddot{a}`	$\dddot{a}$	`\dddot{a}`	$\ddddot{a}$	`\ddddot{a}`
$\widetilde{abc}$	`\widetilde{abc}`			$\widehat{abc}$	`\widehat{abc}`		

Переменные можно снабдить математическими акцентами, представленными в табл. 4.3, например `f_x = m\dot{v}_x` ⇝ $f_x = m\dot{v}_x$. Хотя среди них имеется и значок вектора, `\vec\alpha` ⇝ $\vec\alpha$, его следует использовать только в каких-то специальных случаях, так как *стандартом обозначения векторов является жирный шрифт*. Для обычных латинских букв это прямой шрифт `\mathbf`, а векторы, обозначенные греческими буквами или буквами математических алфавитов, вводят с помощью команд `\bm` или `\boldsymbol`:

```
\mathbf{a} = a_x\mathbf{e}_x
  +a_y\mathbf{e}_y +a_z\mathbf{e}_z,
\xi _x = \boldsymbol\xi \mathbf{e}_x,
\bm{\mathbffrak{H}} =
  \mathbf{H} +\mathbf{H}_d.
```

$$\mathbf{a} = a_x\mathbf{e}_x + a_y\mathbf{e}_y + a_z\mathbf{e}_z,$$
$$\xi_x = \boldsymbol{\xi}\mathbf{e}_x,$$
$$\bm{\mathfrak{H}} = \mathbf{H} + \mathbf{H}_d.$$

В традициях российской полиграфии жирные латинские буквы, обозначающие векторы, должны оставаться прямыми, а индексы не должны

быть жирными. Буквы, печатаемые командами `\mathbf{abc}` ↝ $\mathbf{abc}$ и `\bm{abc}` ↝ $\boldsymbol{abc}$, имеют разное начертание, поэтому для набора векторов не следует применять упрощенные конструкции типа:

```
\xi_x = \bm{\xi e_x},
\bm{\mathfrak{H} = H_a +H_d}.
```

$$\xi_x = \boldsymbol{\xi e_x},$$
$$\boldsymbol{\mathfrak{H} = H_a + H_d}.$$

Математические символы

В табл. 4.4 собрано большое количество математических символов, разбитых по категориям. Большинство из них содержит кодировка unicode, и лишь небольшая часть, выделенная фоном, отсутствует в ней. Для наглядности и компактности команды, печатающие символы, опущены, а вместе они представлены в табл. В.14 – В.18 прил. В.

Среди бинарных операторов, размещенных в верхней части табл. 4.4, наиболее часто используются постоянная Планка $\hbar$ (`\hslash`) или $\hbar$ (`\hbar`), символ бесконечности ∞ (`\infty`), оператор градиента ∇ (`\nabla`), дифференциал частной производной ∂ (`\partial`), а также знаки умножения $\cdot$ (`\cdot`) и $\times$ (`\times`).

Среди соотношений, к которым относится и большой набор различных стрелок,[2] часто используются правая стрелка $\to$ (`\to`), знаки пропорциональности $\propto$ (`\propto`), тождества $\equiv$ (`\equiv`), примерно $\simeq$ (`\simeq`), порядка $\sim$ (`\sim`), не равно $\neq$ (`\neq`), а также различные варианты сравнения: $\leq$ (`\leq`) и $\geq$ (`\geq`), $\lesssim$ (`\lesssim`) и $\gtrsim$ (`\gtrsim`), $\ll$ (`\ll`) и $\gg$ (`\gg`).

Как видно в табл. 4.4, в LaTeX определено большое количество соотношений с отрицанием, а для построения отсутствующих предусмотрена команда `\not{символ}`, накладывающая слэш на стоящий за ней символ, например `\not\equiv` ↝ $\not\equiv$.

В табл. 4.5 собраны различные варианты точек и многоточий, а также символы с двоеточиями, определяемые пакетом mathtools [31]. Печатающие их команды представлены в табл. В.22 и В.23. Математические шрифты Computer Modern содержат все эти символы, кроме последнего диагонального троеточия. Часть из них, выделенная фоном, отсутствует в кодировке unicode.

[2] Отметим, что символы $\nuparrow$ (`\nuparrow`) и $\ndownarrow$ (`\ndownarrow`), добавленные пакетом mathtools, не являются оригинальными, а получены поворотом стрелок $\nleftarrow$ (`\nleftarrow`) и $\nrightarrow$ (`\nrightarrow`). Последние заменяют их при копировании.

Таблица 4.4

Математические символы

Бинарные операторы:

$$\pm\ \mp\ \dotplus\ \lessdot\ \gtrdot\ \curlywedge\ \curlyvee\ \wedge\ \vee\ \barwedge\ \veebar\ \doublebarwedge\ \wr\ \intercal\ \setminus\ \smallsetminus$$
$$\rightthreetimes\ \leftthreetimes\ \dagger\ \triangleleft\ \triangleright\ \cap\ \cup\ \sqcap\ \sqcup\ \Cap\ \Cup\ \uplus\ \cdot\ \centerdot\ *\ \star$$
$$\ltimes\ \rtimes\ \ddagger\ \diamondleft\ \diamondright\ \oplus\ \ominus\ \otimes\ \odot\ \oslash\ \circledast\ \circledcirc\ \circledast\ \circ\ \bullet$$
$$\times\ \divideontimes\ \div\ \trianglelefteq\ \trianglerighteq\ \boxplus\ \boxminus\ \boxtimes\ \boxdot\ \bigtriangleup\ \bigtriangledown\ \diamond\ \amalg$$

Соотношения:

$$\equiv\ \sim\ <\ >\ \leqq\ \geqq\ \subset\ \supset\ \prec\ \succ\ \leqslant\ \geqslant\ \shortmid\ \nmid$$
$$\doteqdot\ \nsim\ \not<\ \not>\ \nleqq\ \ngeqq\ \subseteq\ \supseteq\ \not\prec\ \not\succ\ \lesssim\ \gtrsim\ \mid\ \nmid$$
$$\eqslantless\ \simeq\ \ll\ \gg\ \lneqq\ \gneqq\ \not\subset\ \not\supset\ \preceq\ \succeq\ \lesseqgtr\ \gtrless\ \parallel\ \nparallel$$
$$\doteq\ \backsim\ \lll\ \ggg\ \lneq\ \gneq\ \subsetneq\ \supsetneq\ \precnsim\ \succnsim\ \asymp\ \between\ \shortparallel\ \nshortparallel$$
$$\risingdotseq\ \backsimeq\ \leq\ \geq\ \lnsim\ \gnsim\ \subsetneqq\ \supsetneqq\ \precnsim\ \succnsim\ \frown\ \smile\ \vartriangleleft\ \vartriangleright$$
$$\fallingdotseq\ \eqsim\ \nleq\ \ngeq\ \lnapprox\ \gnapprox\ \nsubseteqq\ \nsupseteqq\ \precnsim\ \succnsim\ \frown\ \smile\ \ntriangleleft\ \ntriangleright$$
$$\neq\ \approx\ \leqslant\ \geqslant\ \lesssim\ \gtrsim\ \subsetneqq\ \supsetneqq\ \lesssim\ \gtrsim\ \vdash\ \dashv\ \ntriangleleft\ \ntriangleright$$
$$\triangleq\ \approxeq\ \nleqslant\ \ngeqslant\ \sqsubset\ \sqsupset\ \subsetneqq\ \supsetneqq\ \lesssim\ \gtrsim\ \models\ \Vdash\ \blacktriangleleft\ \blacktriangleright$$
$$\propto\ \cong\ \leqslant\ \geqslant\ \sqsubseteq\ \sqsupseteq\ \Subset\ \Supset\ \approx\ \approx\ \models\ \models\ \therefore\ \because\ \triangle$$
$$\propto\ \ncong\ \eqcirc\ \circeq\ \in\ \ni\ \bowtie\ \bowtie\ \approx\ \gtrsim\ \nvDash\ \nvDash\ \nvdash\ \nvDash\ \ni$$

Стрелки (соотношения):

$$\leftarrow\ \rightarrow\ \leftrightarrow\ \twoheadleftarrow\ \twoheadrightarrow\ \leftarrowtail\ \rightarrowtail\ \mapsto\ \mapsto\ \leftharpoonup\ \rightharpoonup\ \Leftrightarrow\ \Leftarrow\ \Rightarrow$$
$$\nleftarrow\ \nrightarrow\ \nleftrightarrow\ \leftarrowtail\ \rightarrowtail\ \leftrightarrows\ \rightleftarrows\ \curvearrowleft\ \curvearrowright\ \leftharpoondown\ \rightharpoondown\ \nLeftrightarrow\ \nLeftarrow\ \nRightarrow$$
$$\longleftarrow\ \longrightarrow\ \longleftrightarrow\ \hookleftarrow\ \hookrightarrow\ \Lsh\ \Rsh\ \circlearrowleft\ \circlearrowright\ \leftrightharpoons\ \rightleftharpoons\ \Longleftrightarrow\ \Longleftarrow\ \Longrightarrow$$
$$\uparrow\ \downarrow\ \updownarrow\ \dashleftarrow\ \dashrightarrow\ \nearrow\ \searrow\ \upharpoonleft\ \upharpoonright\ \downharpoonleft\ \downharpoonright\ \Updownarrow\ \Uparrow\ \Downarrow$$
$$\updownarrow\ \updownarrow\ \multimap\ \leftrightsquigarrow\ \rightsquigarrow\ \nwarrow\ \swarrow\ \leftrightsquigarrow\ \rightsquigarrow\ \downharpoonleft\ \downharpoonright\ \curvearrowright\ \Lleftarrow\ \Rrightarrow$$

Различные символы:

$$\partial\ \hbar\ \hbar\ \Im\ \Re\ \wp\ \angle\ \measuredangle\ \sphericalangle\ \square\ \blacksquare\ \lozenge\ \blacklozenge\ \diamondsuit\ \diamond$$
$$\infty\ \imath\ \jmath\ \ell\ \Bbbk\ \mho\ \diagdown\ \diagup\ \triangle\ \triangledown\ \blacktriangle\ \blacktriangledown\ \spadesuit\ \heartsuit\ \clubsuit$$
$$\nabla\ \aleph\ \beth\ \gimel\ \daleth\ \eth\ `\ '\ \exists\ \nexists\ \varnothing\ \emptyset\ \checkmark\ \surd\ \maltese$$
$$\int\ \circledR\ \circledS\ \yen\ \digamma\ \Game\ \bot\ \top\ \complement\ \neg\ \sharp\ \flat\ \natural\ \forall\ \star$$

Таблица 4.5

Точки, многоточия и символы с двоеточием

$x \cdot x$	$x : x$	$x \mathinner{\ldots} x$	$x \cdot x$	$x . x$	$x \mathinner{\ldots} x$	$x \mathinner{\ldots} x$	$, \mathinner{\ldots} x$
$x : x$	$x :: x$	$x \vdots x$	$x \mathinner{\ddots} x$	$x \cdots x$	$+ \cdots x$	$\times \cdots x$	$\int \cdots x$
$x :\!- x$	$x -\!: x$	$x := x$	$x =: x$	$x :\sim x$	$x \sim\!: x$	$x :\approx x$	$x \approx\!: x$
$x -\!:: x$	$x ::\!- x$	$x ::= x$	$x =:: x$	$x ::\sim x$	$x \sim\!:: x$	$x ::\approx x$	$x \approx\!:: x$

4.1.2. Функции и основные конструкции

Как правило, формулы содержат функции и конструкции типа верхних и нижних индексов, корней, дробей и т. п.

Функции выделяются прямым шрифтом и небольшой отбивкой, величина которой меняется в зависимости от представления аргумента и символа, находящегося перед функцией.

Таблица 4.6

Математические функции

$\cos x$	$\arccos x$	$\operatorname{ch} x$	$\cosh x$	$\operatorname{csc} x$	$\operatorname{cosec} x$
$\sin x$	$\arcsin x$	$\operatorname{sh} x$	$\sinh x$	$\sec x$	
$\operatorname{tg} x$	$\operatorname{arctg} x$	$\operatorname{th} x$	$\tanh x$	$\tan x$	$\arctan x$
$\operatorname{ctg} x$	$\operatorname{arcctg} x$	$\operatorname{cth} x$	$\coth x$	$\cot x$	
$\exp x$	$\ln x$	$\lg x$	$\log x$	$\lim x$	
$\arg x$	$\deg x$	$\det x$	$\dim x$	$\gcd x$	$\hom x$
$\inf x$	$\ker x$	$\max x$	$\min x$	$\Pr x$	$\sup x$
$\mathsf{D}\, x$	`\Variance x`		$y \mod x$	`y \mod x`	
$\mathsf{P}\, x$	`\Prob x`		$y \bmod x$	`y \bmod x`	
$y\ (x)$	`y \pod x`		$y\ (\mathrm{mod}\ x)$	`y \pmod x`	

Набор функций, определенных в LaTeX, приведен в табл. 4.6. В ее верхней части собраны функции, наименования которых идентичны именам печатающих их команд, например `\sin x` ⇝ $\sin x$. В нижней части представлены функции, обозначения которых отличаются от имен выводящих их команд. Функции, имеющие специфические левые и правые отбивки, показаны в окружении переменных.

Несмотря на свою обширность набор функций LaTeX не в состоянии охватить все их разнообразие, поэтому авторам предоставлена возможность самим создавать недостающие. Для этого недостаточно набрать

имя функции прямым шрифтом `\mathrm`, сравните

`x \mathrm{sin} x` $\rightsquigarrow x\mathrm{sin}x$ и `x \sin x` $\rightsquigarrow x \sin x$.

Команда `\operatorname{`*функция*`}` создает функцию непосредственно при ее использовании, например `\operatorname{erf} x` $\rightsquigarrow \operatorname{erf} x$.

Для часто используемых функций, воспользовавшись конструкцией

`\DeclareMathOperator{`*команда*`}{`*функция*`}`,

можно создать собственные команды. Она определит новую *команду* для вывода *функции*. Определения новых функций лучше разместить в преамбуле или в начале документа после команды `\begin{document}`.[3] В следующем примере для функции grad вводится команда `\G`:

```
\DeclareMathOperator{\G}{grad}
\begin{document}
   ... x\G(x)/y ...
```

... $x\ \mathrm{grad}(x)/y$ ...

Корни

Команда `\sqrt[`*степень*`]{`*выражение*`}` формирует корень n-й степени, например `\sqrt[4]{xx^*}=\sqrt{\pm|x|}` $\rightsquigarrow \sqrt[4]{xx^*} = \sqrt{\pm|x|}$. Показатель степени квадратного корня опускается, а у подкоренного выражения, состоящего из одного символа, скобки можно опустить: `\sqrt\xi` $\rightsquigarrow \sqrt\xi$. Команды `\leftroot{`*число*`}` и `\uproot{`*число*`}` помогают настроить положение показателя степени, если он смотрится не очень хорошо. Их аргументами являются целые числа. Положительные значения сдвигают показатель немного влево и вверх, а отрицательные — вправо и вниз. Эти команды должны находиться перед показателем в параметре команды `\sqrt`. Сравните, к примеру,

`\sqrt[\xi]{x}` $\rightsquigarrow \sqrt[\xi]{x}$ и $\sqrt[\xi]{x}$ $\leftsquigarrow$ `\sqrt[\leftroot{-1}\xi]{x}`.

Дроби

Команды `\frac{`*числитель*`}{`*знаменатель*`}` верстают дроби. Размер символов *числителя* и *знаменателя* зависит от типа формул. Чтобы строчные формулы не раздвигали строки, в дробях используется мелкий шрифт `$\frac{1+x}{1+x}$` $\rightsquigarrow \frac{1+x}{1+x}$, а в дробях, находящихся в числителе или знаменателе, он становится еще меньше:

`$x+\frac{x+1}{x+\frac{x+1}{x +\frac{1}{x +1}}}$` $\rightsquigarrow$ $x+\frac{x+1}{x+\frac{x+1}{x+\frac{1}{x+1}}}$.

[3] Издатель может полностью заменить преамбулу документа, и тогда введенные в ней определения пропадут.

В дробях вынесенных формул шрифт не уменьшается:

```
$$\frac{1+x}{1+x}$$
```
$$\frac{1+x}{1+x}.$$

Команды `\tfrac{•}{•}` и `\dfrac{•}{•}` позволяют регулировать размер дробей. Их числители и знаменатели имеют такой же размер, как у дроби `\frac`, находящейся, соответственно, в строчной или вынесенной формуле (о размерах символов см. разд. 4.2). При их использовании предыдущий пример будет смотреться намного лучше:

```
x +\tfrac{x+1}{x +\tfrac{x+1}{x
                +\tfrac{1}{x +1}}}
```
$$x+\tfrac{x+1}{x+\tfrac{x+1}{x+\tfrac{1}{x+1}}},$$

```
x +\dfrac{x+1}{x +\dfrac{x+1}{x
                +\dfrac{1}{x +1}}}
```
$$x+\dfrac{x+1}{x+\dfrac{x+1}{x+\dfrac{1}{x+1}}}.$$

Для набора вложенных дробей предназначена специальная команда `\cfrac{•}{•}`. Выводя дробь размером `\dfrac`, она оптимизирует вертикальную отбивку знаменателя:

```
x +\cfrac{x+1}{x +\cfrac{x+1}{x
                +\cfrac{1}{x +1}}}
```
$$x+\cfrac{x+1}{x+\cfrac{x+1}{x+\cfrac{1}{x+1}}}.$$

Если числитель много длинней знаменателя, или наоборот, команды

```
\splitfrac{верхняя строка}{нижняя строка},
\splitdfrac{верхняя строка}{нижняя строка},
```

определяемые пакетом mathtools, позволяют разбить длинное выражение на две строки. Их действия различаются величиной межстрочного интервала, который больше в случае `\splitdfrac`. Сравните:

```
\[ a = \frac{\splitfrac{xxxxxxxxxx+}{+yyyyyyyyyy}}{z} =
       \frac{\splitdfrac{xxxxxxxxxx+}{+yyyyyyyyyy}}{z} \].
```
$$a=\frac{\splitfrac{xxxxxxxxxx+}{+yyyyyyyyyy}}{z}=\frac{\splitdfrac{xxxxxxxxxx+}{+yyyyyyyyyy}}{z}.$$

Если строки по-прежнему остаются слишком длинными, разбейте их на более мелкие части и используйте вложенные команды:

```
\[ a = \frac{
     \splitfrac{
\splitfrac{xxxxxxxx+}{+yyyyyyyy+}
     }{+zzzzzzzz}
             }{b} \]
```

$$a = \frac{\splitfrac{\splitfrac{xxxxxxxx+}{+yyyyyyyy+}}{+zzzzzzzz}}{b}$$

Индексы

Команды `^{выражение}` и `_{выражение}` формируют верхние и нижние индексы, например `x_i^2` ⇝ x_i^2. Порядок следования индексов произволен, а их вложенность не ограничена, но размер индексов уменьшается только до второго «уровня», начиная с которого он остается неизменным: `x^{2^{2^2}}` ⇝ $x^{2^{2^2}}$. При наборе следует различать индексы индексированного выражения, `{x^2_i}^2_j` ⇝ $x_{i\,j}^{2\,2}$, и индексы индексов: `x^{2^2}_{i_j}` ⇝ $x_{i_j}^{2^2}$. Их различие, в частности, не позволяет использовать двойные индексы одного типа:

`x\_i\_j` ⇝ **Ошибка!** ↜ `x\^{ }2\^{ }2`.

Положение индексов определяется высотой и глубиной индексируемого выражения. Сравните выражения, в которых индексы α и β привязаны к закрывающей скобке, `(x_i^2)^\alpha_\beta` ⇝ $(x_i^2)_\beta^\alpha$, и к выражению в целом, `{(x_i^2)}^\alpha_\beta` ⇝ $(x_i^2)_\beta^\alpha$.

Индексы ассоциируются с выражением, стоящим слева, и потому «подтягиваются» к нему. Интуитивные попытки сформировать левые индексы к выражению, стоящему справа, не дают желаемый результат:

`f ^3_3L_3^3, f {^3_3L^3_3}, f {}^3_3L^3_3` ⇝ $f_3^3L_3^3$.

Все три представленных варианта кода верстаются одинаково. Несмотря на пробел между f и последующими индексами и отсутствие такового между ними и L, видно, что индексы привязаны к левому символу, а группирующие скобки не помогают скорректировать их положение. Решает задачу тонкий пробел, вставленный между f и индексируемым выражением: `f\,^{3}_{3}{L}_3^{3}` ⇝ $f_3^3L_3^3$.

Для расстановки левых индексов пакет mathtools вводит команду

`\prescript{`*верхний индекс*`}{`*нижний индекс*`}{`*выражение*`}` .

Ее первым и вторым аргументом являются индексы, а в последний аргумент помещается индексируемое выражение. Вот как она формирует нашу конструкцию: `f\prescript{3}{3}{L}_3^3` ⇝ $f_3^3L_3^3$. Если выражение имеет также и правые индексы, как в нашем случае, не стоит вносить их в аргумент. Это вызовет избыточное вертикальное смещение левых индексов: `f\prescript{3}{3}{L_3^3}` ⇝ $f_3^3L_3^3$.

4.1.3. Большие операторы и функции с пределами

Отдельную группу объектов образуют большие операторы и функции, имеющие пределы, собранные в табл. 4.7 и В.21. Среди них оператор `\bigtimes` ⇝ $\bigtimes$ вводит пакет mathtools.

Таблица 4.7

Большие операторы и функции, имеющие пределы

$\sum\sum$	$\bigcap\bigcap$	$\bigsqcap\bigsqcap$	$\biguplus\biguplus$	$\bigoplus\bigoplus$	$\int\int$	$\iint$ $\iiint$
$\prod\prod$	$\bigcup\bigcup$	$\bigsqcup\bigsqcup$	$\bigtimes\bigtimes$	$\bigotimes\bigotimes$	$\oint\oint$	$\iiint\iiint$
$\coprod\coprod$	$\bigvee\bigvee$	$\bigwedge\bigwedge$		$\bigodot\bigodot$	$\iint\iint$	$\int\cdots\int\int\cdots\int$
$\det x$	$\gcd x$	$\inf x$	$\overline{\lim}\, x$	$\limsup x$	$\varinjlim x$	$\operatorname{inj\,lim} x$
$\max x$	$\min x$	$\Pr x$	$\underline{\lim}\, x$	$\liminf x$	$\varprojlim x$	$\operatorname{proj\,lim} x$
			$\lim x$			

Прежде всего отметим, что операторы имеют разные размеры в вынесенных и строчных формулах, что показано в таблице для каждого из них. Особенности их поведения рассмотрим на примере интеграла `\int` ⇝ $\int$, суммы `\sum` ⇝ $\sum$ и функции предела `\lim` ⇝ $\lim$. Часто они имеют пределы, для расстановки которых используются команды индексов: `\sum_{i=1}^\infty` ⇝ $\sum_{i=1}^{\infty}$, `\int_a^b` ⇝ $\int_a^b$. В вынесенных формулах пределы могут располагаться либо сбоку, как у определенных интегралов, либо сверху и снизу, как у сумм и пределов:

`\int_a^b` ⇝ $\displaystyle\int_a^b$, `\sum_{i=0}^\infty` ⇝ $\displaystyle\sum_{i=0}^{\infty}$, `\lim_{x\to0}` ⇝ $\displaystyle\lim_{x\to0}$.

В строчных формулах, чтобы не раздвигать строки, пределы по умолчанию проставляются сбоку: $\int_a^b$, $\sum_{i=0}^{\infty}$, $\lim_{x\to0}$.

Команды `\limits` и `\nolimits` управляют положением индексов. Они ставятся между оператором и его пределами. Использование `\limits` гарантирует расстановку пределов над и под оператором, а команда `\nolimits`, напротив, всегда размещает их сбоку:

`\int\limits_a^b` ⇝ $\displaystyle\int\limits_a^b$, `\sum\nolimits_{i=0}^\infty` ⇝ $\displaystyle\sum\nolimits_{i=0}^{\infty}$.

Команда `\substack{`*строка1*`\\` *строка2*`...}` формирует пределы, состоящие из нескольких строк, которые иногда требуются в конструкциях с большими операторами. Строки разделяются командой `\\` и центрируются:

```
\sum_{\substack{2<i<m,\\ j<10}} P(i,j)
```
$\rightsquigarrow$ $\displaystyle\sum_{\substack{2<i<m,\\ j<10}} P(i,j).$

Окружение `subarray` позволяет сверстать многострочные индексы с любым выравниванием. Оно имеет аргумент, принимающий значение: `c` — центрирование, `l` или `r` — выравнивание слева или справа:

```
\sum_{\begin{subarray}{l}
  2<i<m,\\ j<10
\end{subarray}} P(i,j).
```
$\displaystyle\sum_{\substack{2<i<m,\\ j<10}} P(i,j).$

У некоторых операторов пределы могут находиться как слева, так и справа. Расставить их помогает конструкция:

`\sideset{`*левые индексы*`}{`*правые индексы*`}{`*оператор*`}`.

Ее аргументами являются команды расстановки индексов и большой оператор. Независимо от типа формулы она всегда печатает оператор большого размера: `$\sideset{^1_2}{^3_4}\prod$` $\rightsquigarrow$ $\displaystyle\sideset{^1_2}{^3_4}\prod$.

Для создания функций, имеющих пределы, определены команды

`\operatorname*{`*функция*`}`,
`\DeclareMathOperator*{`*команда*`}{`*функция*`}`.

Подобно функциям из нижней части табл. 4.7 они имеют правые пределы, положение которых регулируется автоматически и при необходимости корректируется командами `\limits` и `\nolimits`, например

```
$ \operatorname*{F}
  \limits_{x\to 0}^{y\to x}
  (x,y,z) $
```
$\displaystyle\operatorname*{F}_{x\to 0}^{y\to x}(x,y,z).$

Данный пример показывает, что у создаваемой функции, равно как и у всех функций из табл. 4.7, определен не только нижний, но и верхний предел.

4.1.4. Разделители

Особый тип объектов составляют разделители. Так как они привычно ассоциируются со скобками, будем считать их синонимами. Наряду с круглыми, квадратными и фигурными скобками очень часто исполь-

зуются угловые скобки: `\langle x \rangle` ⇝ $\langle x\rangle$. Полный набор разделителей приведен в табл. 4.8 и В.13.

Таблица 4.8

Разделители

(	)	[	]	/	\	|	‖	⌈	⌉	⌜	⌝	↑	↓	↕
{	}	⟨	⟩	|	|	‖	‖	⌊	⌋	⌞	⌟	⇑	⇓	⇕

Вдобавок к скобкам, показанным в табл. 4.8, набор разделителей содержит команды `\lvert`, `\rvert` ⇝ | и `\lVert`, `\rVert` ⇝ ‖, генерирующие левые и правые знаки модуля и нормы. Их отбивка отличается от отбивки обычного знака модуля |, водимого с клавиатуры, и знака нормы ‖, печатаемого командой `\|`, сравните

```
\sin\lvert x\rvert +\cos \lVert f\rVert   ⇝  $\sin|x| + \cos\|f\|$,
\sin|x| +\cos \|f\|                       ⇝  $\sin |x| + \cos \|f\|$.
```

Размер скобок должен быть согласован с высотой заключенного в них выражения. Настроить его позволяет набор команд

`\big`,	`\bigl`,	`\bigr`,	`\bigm`,
`\Big`,	`\Bigl`,	`\Bigr`,	`\Bigm`,
`\bigg`,	`\biggl`,	`\biggr`,	`\biggm`,
`\Bigg`,	`\Biggl`,	`\Biggr`,	`\Biggm`,

которые ставятся перед скобками и увеличивают их размер от `\big` до `\Bigg`. Покажем пример обычных скобок и все варианты их увеличения:

```
\Biggl(\biggl(\Bigl(\bigl(
  (t+y)
\bigr)\Bigr)\biggr)\Biggr)
```

$$\Biggl(\biggl(\Bigl(\bigl((t+y)\bigr)\Bigr)\biggr)\Biggr).$$

Команды с именами, заканчивающимися на `l`, `r` и `m`, обеспечивают дополнительную отбивку вокруг разделителей. Они предназначены, соответственно, для левых и правых скобок и символов, разделяющих две части выражения. Сравните:

`\frac{x}{y} \big/ \frac...` ⇝ $\frac{x}{y}\big/\frac{y}{x}$, $\frac{x}{y}\bigm/\frac{y}{x}$ ↜ `\frac...\bigm/ \frac...`

С настройкой скобок вручную связано две проблемы. Во-первых, их максимальный размер ограничен, тогда как заключенные в них выражения могут оказаться более громоздкими. Во-вторых, высота выражения может различаться в строчной и вынесенной формулах, и если размер скобок зафиксирован, конструкция в них будет выглядеть по-

разному. Например, дробь в скобках, неплохо смотревшаяся в строчной формуле, `\bigl(\frac{b}{y}\bigr)` ↝ $\bigl(\frac{b}{y}\bigr)$, оказавшись в вынесенной, станет непрезентабельной: $\Bigl(\frac{b}{y}\Bigr)$. Обе проблемы устраняет конструкция

`\left`• *выражение* `\right`•.

Команды `\left` и `\right`, которые ставятся перед разделителями обозначенными символами •, служат метками границ *выражения*, высоту которого компилятор должен вычислить, чтобы поставить скобки нужного размера. Данная конструкция генерирует разделители, размер которых не ограничен. Используем ее в нашем примере: `\left(\frac{b}{y}\right)` ↝ $\left(\frac{b}{y}\right)$ — строчная, $\left(\frac{b}{y}\right)$ — вынесенная.

Команды `\left` и `\right` используются только в паре. В формулах, имеющих сложную конструкцию, они должны находиться в одном и том же поле одной и той же строки (см. разд. 4.3 и 4.4). Разбиение пары командой перевода строки `\\` или знаком разделения полей `&` вызовет ошибку компиляции. Если скобки охватывают выражение, состоящее из нескольких строк (или полей), нужно использовать пару конструкций `\left`• и `\right`•. В первой указывается левый разделитель, а правый заменяется точкой, а во второй, наоборот, левый заменяется точкой, а правый указывается:

```
\begin{multline*}
  \left[\frac{1}{2\ch(R/r)}
    + ...\right.\\
  \left. + ...
    \frac{1}{2\sh(R/r)} \right].
\end{multline*}
```

$$\left[\frac{1}{2\,\mathrm{ch}(R/r)} + \dots + \dots \frac{1}{2\,\mathrm{sh}(R/r)}\right].$$

Поясним сказанное на примере представленного выше выражения, состоящего из двух строк и охваченного скобками. В каждой строке пары `\left`• и `\right`• содержат по одной скобке и одной точке. Открывающую скобку `\left[` «закрывает» команда `\right.` в конце первой строки, а начинающая вторую строку команда `\left.` «закрывается» скобкой `\right]`.

4.1.5. Сверху и снизу

В формулах некоторые объекты зачастую располагаются один над другим на нескольких уровнях, как числители и знаменатели в дробях. Аналогично дробям устроены конструкции

```
\binom{верхняя строка}{нижняя строка},
\tbinom{верхняя строка}{нижняя строка},
\dbinom{верхняя строка}{нижняя строка},
```

отличающиеся лишь отсутствием разделительной черты и круглыми скобками, охватывающими сверстанное выражение. Как следует из имени команд, основным их назначением является печать биномиальных коэффициентов:

```
\dbinom{n}{k}=\dfrac{n!}{(n-k)!\,k!}
```
$\rightsquigarrow$ $\dbinom{n}{k} = \dfrac{n!}{(n-k)!\,k!}$.

Размер символов конструкции `\binom` зависит от контекста, в котором она находится, а в конструкциях `\tbinom` и `\dbinom` он соответствуют размеру символов основной строки строчных и вынесенных формул.

Еще одна конструкция задает *условие* выполнения *соотношения*:

```
\stackrel{условие}{соотношение}
```

Условие верстается размером индексов, а символ соотношения печатается обычным шрифтом и центрируется относительно осевой линии:

```
\mathbf{v} \stackrel{\mathrm{def}}\equiv
(v_x,v_y,v_z)
```
$\mathbf{v} \stackrel{\mathrm{def}}{\equiv} (v_x, v_y, v_z)$.

Чтобы не загромождать формулу скобками, в примере использовано правило наследования аргументов.

Разместить практически что угодно над или под *выражением* позволяют конструкции

```
\overset{вверху}{выражение}, \underset{внизу}{выражение},
\overunderset{вверху}{внизу}{выражение}.
```

Основное выражение набирается обычным шрифтом и привязывается к базовой линии, а размер символов верхних и нижних выражений соответствует индексам, но его легко настроить декларациями, обсуждаемыми в разд. 4.2:

```
\underset{\textstyle X\notin \mathbb{P}}{X}
\bigotimes
\underset{\textstyle Y\in \mathbb{P}}{Y}
```
$\underset{\textstyle X\notin \mathbb{P}}{X} \bigotimes \underset{\textstyle Y\in \mathbb{P}}{Y}$.

На с. 157 обсуждается создание конструкции, действующей аналогично `\overset` и `\underset`, но привязанной не к базовой, а к осевой линии строки.

Ряд команд рисует над или под *выражениями* стрелки, согласуя их длину с длиной выражений.

`\overleftarrow{выражение}`, `\underleftarrow{выражение}`,
`\overrightarrow{выражение}`, `\underrightarrow{выражение}`,
`\overleftrightarrow{выраж.}` `\underleftrightarrow{выраж}`.

Выражения печатаются в строке, а стрелки опускаются или поднимаются над ней. Команды, представленные в левой колонке, печатают верхние стрелки, а в правой — нижние:

`\overleftarrow{x>y},\ \underrightarrow{x<y}` $\rightsquigarrow$ $\overleftarrow{x>y},\ \underrightarrow{x<y}$.

Стрелки другого набора печатаются на уровне осевой линии строки, а над и под ними выводятся условные выражения.

`\xleftarrow[снизу]{сверху}`, `\xrightarrow[снизу]{сверху}`.

Условные выражения имеют размер индексов:

```
AT \xleftarrow[T>0]{i>2}  B_i(T)
   \xrightarrow[T<0]{i=5} C/T
```

$AT \xleftarrow[T>0]{i>2} B_i(T) \xrightarrow[T<0]{i=5} C/T.$

Пакет mathtools вводит аналогичные команды для большого количества других стрелок:

`\xleftrightarrow[снизу]{сверху}` $\xleftrightarrow[снизу]{сверху}$,

`\xlongleftarrow[•]{•}`	$\xlongleftarrow[\cdots]{\cdots}$,	`\xlongrightarrow[•]{•}`	$\xlongrightarrow[\cdots]{\cdots}$,
`\xmapsto[•]{•}`	$\xmapsto[\cdots]{\cdots}$,	`\xLeftrightarrow[•]{•}`	$\xLeftrightarrow[\cdots]{\cdots}$,
`\xLeftarrow[•]{•}`	$\xLeftarrow[\cdots]{\cdots}$,	`\xRightarrow[•]{•}`	$\xRightarrow[\cdots]{\cdots}$,
`\xhookleftarrow[•]{•}`	$\xhookleftarrow[\cdots]{\cdots}$,	`\xhookrightarrow[•]{•}`	$\xhookrightarrow[\cdots]{\cdots}$,
`\xleftharpoondown[•]{•}`	$\xleftharpoondown[\cdots]{\cdots}$,	`\xrightharpoondown[•]{•}`	$\xrightharpoondown[\cdots]{\cdots}$,
`\xleftharpoonup[•]{•}`	$\xleftharpoonup[\cdots]{\cdots}$,	`\xrightharpoonup[•]{•}`	$\xrightharpoonup[\cdots]{\cdots}$,
`\xrightleftharpoons[•]{•}`	$\xrightleftharpoons[\cdots]{\cdots}$,	`\xleftrightharpoons[•]{•}`	$\xleftrightharpoons[\cdots]{\cdots}$

Команды `\overline{выражение}` и `\underline{выражение}` используются для выражений отчеркивания сверху и снизу:

`\overline{x+y}\stackrel?\ne\underline{x+y}` $\rightsquigarrow$ $\overline{x+y} \stackrel{?}{\ne} \underline{x+y}$,

а команды `\overbrace{выражение}` и `\underbrace{выражение}` позволяют охватить их фигурными скобками. Для них определены индексы, которые выводятся в виде меток к скобкам:

`\underbrace{\overbrace{x+a+b}^{12}+y}_{>100}` $\rightsquigarrow$ $\underbrace{\overbrace{x+a+b}^{12}+y}_{>100},$

`\underbracket{\overbracket{x+a+b}^{12}+y}_{>100}` $\rightsquigarrow$ $\underbracket{\overbracket{x+a+b}^{12}+y}_{>100}.$

Пакет mathtools добавляет к ним показанные выше квадратные скобки:

`\overbracket[`*толщина линии*`][`*высота*`]{`*выражение*`}` ,
`\underbracket[`*толщина линии*`][`*высота*`]{`*выражение*`}` .

По умолчанию *толщина линии* и *высота* выступа скобок привязываются к размеру шрифта и составляют, соответственно, (5/18) ex и 0.7 ex, но параметры команд позволяют настроить их по своему усмотрению.

4.1.6. Текст, пробелы, фантомы и прочее

Хотя русские буквы непосредственно в формулах не определены, вставить текст в формулу очень просто. Для этого можно воспользоваться любой командой, начинающейся словом `\text...`, из табл. 2.6, например: `abcd\ne\textit{абвг}` $\rightsquigarrow$ *abcd* $\neq$ *абвг*. Следует учитывать, что шрифт текста, окружающего формулу и вставляемого в нее, не должен меняться, если на то нет особых причин. Идентичность шрифтов обеспечивает команда `\text{`*текст*`}` , печатающая *текст* шрифтом, использовавшимся перед формулой:

`{\sf греческие буквы $\pi\text{ и }\xi$}` $\rightsquigarrow$ {\sf греческие буквы $\pi\text{ и }\xi$}.

Как говорилось ранее, обычные пробелы в формулах полностью игнорируются, их можно поставить только с помощью команд. Для печати пробелов заданной ширины используются команды, перечисленные в табл. 2.3. Команды `\hspace{`*длина*`}` и `\mspace{`*длина*`}` позволяют регулировать ширину пробела. В первой длину можно задать в любых единицах, а во второй — только в единицах `mu` = 1/18 em. Как и в тексте, положительная длина раздвигает элементы формул, а отрицательная сближает их.

В тонкой настройке формул помогают «фантомы», незримо участвующие в вычислении размеров:

`\phantom{`*выражение*`}` ,
`\hphantom{`*выражение*`}` , `\vphantom{`*выражение*`}` .

Эти команды могут использоваться не только в формулах, но и в тексте, их аргументы — математические выражения или текст. Команда

\hphantom печатает пробел с шириной выражения; \vphantom вставляет «распорку», имеющую высоту и глубину выражения. И наконец, \phantom вставляет пустой бокс с полными размерами выражения. Покажем использование фантомов на следующем примере:

```
\sqrt{a} + \sqrt{d} + \sqrt{y}
```
$\rightsquigarrow$ $\sqrt{a} + \sqrt{d} + \sqrt{y}.$

Сумма корней смотрится лучше, если их размеры одинаковы:

```
\sqrt{a\vphantom{dy}} + \sqrt{d\vphantom{y}}
+ \sqrt{y\vphantom{d}}
```
$\sqrt{a} + \sqrt{d} + \sqrt{y}.$

В нашем случае это обеспечивают фантомы, корректирующие высоту и глубину подкоренных выражений, но хороший результат достигается с помощью довольно громоздких конструкций. Их можно заменить одной командой \mathstrut $\equiv$ \vphantom{(}, вставляющей распорку с высотой и глубиной круглой скобки:

```
\sqrt{a\mathstrut} + \sqrt{d\mathstrut}
+ \sqrt{y\mathstrut}
```
$\sqrt{a} + \sqrt{d} + \sqrt{y}.$

Противоположно фантомам действует пара команд

\lefteqn{выражение} и \smash[размер]{выражение}.

Первая из них печатает выражение и обнуляет его длину, поэтому следующие за ним символы печатаются поверх самого выражения, например \lefteqn\times + $\rightsquigarrow$ $\times$. Команда \smash, печатая выражение, обнуляет его вертикальные размеры, например {\smash{\Big|}}_1^ $\rightsquigarrow$ $\Big|_1^2$. Видно, что положение индексов не соответствует размерам вертикальной линии, к которой они привязаны. Команда \smash имеет параметр, принимающий значения t и b, позволяющие обнулить либо только высоту, либо только глубину выражения. Сравните:

{\smash[b]{\Big|}}_1^2* $\rightsquigarrow$ $\Big|_1^2$, $\Big|_1^2$ $\leftsquigarrow$ {\smash[t]{\Big|}}_1^2.

Фигурные скобки вокруг конструкции \smash не являются избыточными, так как необходимы для правильной расстановки индексов.

Разберем поучительный пример совместного действия фантома и команды \lefteqn из книги [2]:

```
\lefteqn{\overbrace{
  \phantom{1+2+3} }^6}
 1+\underbrace{2+3+4}_9
```
$\overbrace{1+\underbrace{2+3}_{9}+4}^{6}$

В этой конструкции фигурные скобки, охватывающие разные части последовательности цифр, перекрываются. При непосредственном использовании команд \overbrace и \underbrace цифры 2 и 3

будут напечатаны дважды, а скобки не перекроются, поэтому команда `\overbrace`, выводящая верхнюю скобку, размещена в аргументе команды `\lefteqn`. Длину скобки устанавливает помещенный в ее аргумент фантом «1+2+3». Команда `\lefteqn` сдвигает вывод самого выражения к левому краю скобки, и три его первые цифры, воспроизводящие фантом, аккуратно вписываются под нее. Нижнюю скобку выводит команда `\underbrace`, охватывающая другую часть цифр.

4.2. Размер символов в формулах

Формула является частью текста, поэтому размер ее символов согласуется с его текущим размером. Чтобы увеличить или уменьшить формулу, нужно изменить размер шрифта *перед* ней, например, `{\footnotesize $x=y$}` ⇝ $x=y$ и `{\Large $x=y$}` ⇝ $x=y$. Это справедливо как для строчных, так и вынесенных формул.[4]

В самих формулах определены всего три размера: T — для основных символов, S — для индексов, и SS — для вторых индексов. Их величины вычисляются исходя из текущего размера текстового шрифта. В строчных формулах, находящихся среди текста обычного размера, они примерно соответствуют размерам `\normalsize`, `\scriptsize` и `\tiny`. Четыре комбинации расстановки размеров символов в дробях и индексах образуют математические стили.

`\displaystyle` — стиль вынесенных формул. Символы, стоящие в основной строке формулы и в дробях, имеют одинаковый размер T. Индексы имеют размер S. Таким же размером набираются дроби, стоящие в числителе и знаменателе. Индексы индексов, дроби в индексах, а также индексы и дроби во «вложенных дробях» имеют размер SS:

$$2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2+x}}}}.$$

`\textstyle` — стиль формул в строке. Символы основной строки формулы имеют размер T. Индексы и дроби набираются размером S. Размер остальных символов равен SS: $2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2+x}}$.

[4] В окружении equation декларацию, изменяющую размер шрифта, можно разместить внутри непосредственно перед формулой.

`\scriptstyle` — стиль индексов. Символы основной строки формулы имеют размер S, а остальные размер SS: $2^{\frac{x}{2}}x^{2^x}+\frac{2^{\frac{x}{2}}x^{2^x}}{2+x}$.

`\scriptscriptstyle` — стиль вторых индексов. Все символы имеют размер SS: $_{2^{\frac{x}{2}}x^{2^x}}$.

При размере текстового шрифта `\tiny` в формулах остается только два размера символов. Более крупный шрифт используется в стилях `\displaystyle` и `\textstyle` для символов, стоящих в основной строке формулы, все же остальные символы, включая символы стилей `\scriptstyle` и `\scriptscriptstyle`, имеют более мелкий размер.

Декларации `\displaystyle`, `\textstyle`, `\scriptstyle` и `\scriptscriptstyle` дают возможность настроить размеры отдельных частей формул. Напомним также, что размер символов числителя и знаменателя удобно регулировать командами `\tfrac` и `\dfrac`, генерирующими дроби стилей `\textstyle` и `\displaystyle` (см. разд. 4.1.2).

Разберем использование математических стилей на примере дроби, стоящей в показателе экспоненты:

```
\mathrm{e}^{ -\frac{\varepsilon}{kT} } ⇝ e^{-\frac{\varepsilon}{kT}}.
```

Стоящая в числителе буква ε оказывается слишком малой в размере SS. Используя команду `\tfrac`, дробь можно увеличить до размера S: $\mathrm{e}^{-\frac{\varepsilon}{kT}}$, но тогда формула раздвигает строки, и знаменатель дроби выглядит слишком большим. Формула станет гораздо лучше, если подстроить только размер символа ε:

```
\mathrm{e}^{ -\frac{\scriptstyle \varepsilon}{kT} } ⇝ e^{-\frac{\varepsilon}{kT}}.
```

Обратите внимание, что число $\mathrm{e} = 2.71828\ldots$, как и все другие константы, принято набирать шрифтом `\mathrm`, так как оно не является переменной «e», например зарядом электрона.

4.3. Математические блоки

В дополнение к перечисленным ранее основным элементам и конструкциям в верстке формул со сложной структурой применяются математические блоки. Их можно использовать как в вынесенных, так и в строчных формулах. Например, блок cases позволяет сверстать строчную формулу, состоящую из двух и более строк,

$|\sin\alpha| = \begin{cases} \sin\alpha, & 0 \le \alpha \le \pi, \\ -\sin\alpha, & \pi \le \alpha \le 2\pi, \end{cases}$ правда она окажется очень громоздкой и будет плохо смотреться внутри абзаца.

Компилятор обрабатывает блоки в математической моде, но сами они ее не инициируют и потому должны использоваться только внутри строчных формул или окружений, создающих вынесенные формулы. Сколь бы сложную структуру ни имел блок, он может находиться в ней в любом месте формулы, где может стоять хотя бы один математический символ. Сверстанный блок становится боксом определенной высоты, глубины и ширины, который компилятор встраивает в формулу на общих основаниях.

Блоки представляют собой окружения со структурой типа таблицы tabular. Каждая строка содержит выровненные поля (ячейки), которые разделяются знаком &, а сами строки разделяются командами \\. Одинаковые поля разных строк образуют одинаково выровненный столбец (колонку) и связываются общей вертикалью выравнивания.

Используя положительную или отрицательную *длину* в параметре команды \\[*длина*], можно сдвигать и раздвигать строки. В формулах команда перевода строки определена только в части окружений, а ее использование вне этих окружений вызовет ошибку компиляции.

В приведенных далее примерах мы пометили поля формул и тип их выравнивания. Конструкцией « ▷|◁ » обозначено центрирование. Знаками « →| » « |← » показаны поля с правым и левым выравниванием. Общая вертикаль, образованная столбцами с правым и левым выравниванием, обозначена как « →||← ». Промежутки между столбцами, вычисляемые автоматически, обозначены отрезками « ⊢⊣ », промежутки, заданные автором, — отрезками « |⟷| », а промежутки, имеющие максимальную ширину, иммитируют расширение: « |◁ ▷| ». Знаком & помечены все столбцы, кроме первого, создаваемого автоматически.

Таблица 4.9

Математические блоки

aligned	alignedat	matrix*	smallmatrix*
array	multlined	bmatrix*	bsmallmatrix*
gathered	lgathered	Bmatrix*	Bsmallmatrix*
split	rgathered	pmatrix*	psmallmatrix*
cases*	dcases*	vmatrix*	vsmallmatrix*
drcases*	rcases*	Vmatrix*	Vsmallmatrix*

В табл. 4.9 перечислены окружения, формирующие математические блоки. Окружения с именами, заканчивающимися звездочкой, имеют по две модификации, например cases и cases*. Их различие подробно обсуждается далее для каждого блока.

Матрицы

Матрицы:

matrix,	matrix*,	smallmatrix,	smallmatrix*,
pmatrix	pmatrix*	psmallmatrix,	psmallmatrix*,
Bmatrix,	Bmatrix*,	Bsmallmatrix,	Bsmallmatrix*,
bmatrix,	bmatrix*,	bsmallmatrix,	bsmallmatrix*,
Vmatrix,	Vmatrix*,	Vsmallmatrix,	Vsmallmatrix*,
vmatrix,	vmatrix*,	vsmallmatrix,	vsmallmatrix*.

```
\begin{matrix}
  x & a+b \\
  0 & y
\end{matrix}
```

$$\begin{matrix} x & a+b \\ 0 & y \end{matrix}$$

```
\begin{pmatrix}
  x & a+b \\
  0 & y
\end{pmatrix}
```

$$\begin{pmatrix} x & a+b \\ 0 & y \end{pmatrix}$$

```
\begin{bmatrix}
  x & a+b \\
  0 & y
\end{bmatrix}
```

$$\begin{bmatrix} x & a+b \\ 0 & y \end{bmatrix}$$

```
\begin{Bmatrix}
  x & a+b \\
  0 & y
\end{Bmatrix}
```

$$\begin{Bmatrix} x & a+b \\ 0 & y \end{Bmatrix}$$

```
\begin{vmatrix}
  x & a+b \\
  0 & y
\end{vmatrix}
```

$$\begin{vmatrix} x & a+b \\ 0 & y \end{vmatrix}$$

```
\begin{Vmatrix}
  x & a+b \\
  0 & y
\end{Vmatrix}
```

$$\begin{Vmatrix} x & a+b \\ 0 & y \end{Vmatrix}$$

Значительный набор блоков составляют матрицы, верстаемые в виде таблиц с несколькими столбцами, разделенными небольшими одинаковыми промежутками. По умолчанию число столбцов ограничено десятью, но, переопределив значение счетчика MaxMatrixCols (см. разд. 7.4), это ограничение можно снять. Ячейки матриц набираются в стиле `\textstyle`.

Окружения, верстающие матрицы, заключенные в скобки, определены пакетом amsmath. Он также вводит окружение matrix, которое в комбинации с конструкцией

`\left•\begin{matrix} ... \end{matrix}\right•`,

позволяет охватить матрицу любыми скобками из табл. 4.8.

Матрицы имеют большой размер, поэтому они обычно размещаются в вынесенных формулах, а для строчных предусмотрены окружения с именами, содержащими слово small. В них используются мелкие шрифты и малые промежутки между столбцами и строками, например `\begin{vsmallmatrix}...` $\rightsquigarrow \left|\begin{smallmatrix} x & 0 \\ 1 & y \end{smallmatrix}\right|$. Пакет amsmath определяет блок smallmatrix, верстающий маленькую матрицу без скобок. Матрицы в скобках добавлены пакетом mathtools.

По умолчанию столбцы матриц центрируются. Чтобы снять это ограничение mathtools вводит окружения с именами, оканчивающимися звездочкой, имеющие параметр, принимающий значения l или r, устанавливающие левое или правое выравнивание столбцов, например

`\begin{psmallmatrix*}[l]`.

С помощью команды `\hdotsfor[`*множитель*`]{`*число столбцов*`}` ячейки матриц можно заполнить точками. Число заполняемых ячеек указывается в ее аргументе. Параметр, задаваемый положительным действительным числом, регулирует разреженность точек. Значения меньше или больше единицы делают последовательность точек более или менее плотной:

```
\begin{pmatrix}
   1 &  2 &  3 & 4 \\
   \hdotsfor{4}    \\
   1 & \hdotsfor[.5]{2} & 4
\end{pmatrix}
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ \dots & \dots & \dots & \dots \\ 1 & \dots & \dots & 4 \end{pmatrix}$$

В маленьких матрицах команда `\hdotsfor` не работает, но в них хорошо смотрятся троеточия из табл. 4.5:

```
\begin{psmallmatrix}
   1 &  \cdots & 9\\
   \vdots & \ddots & \vdots\\
   1 & \cdots & 9
\end{psmallmatrix}.
```

$$\left(\begin{smallmatrix} 1 & \cdots & 9 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 9 \end{smallmatrix}\right).$$

---

array — блок с произвольным выравниванием столбцов

```
\left(  \begin{array}{lcr}
    xy & a+b & c+d \\
\hdotsfor[1.5]{3} \\
    1  & 0 & z \\
\end{array}  \right)
```

$$\left(\begin{array}{lcr} xy & a+b & c+d \\ \dots & \dots & \dots \\ 1 & 0 & z \end{array}\right)$$

`\begin{array}[`*привязка*`]{`*шаблон строки*`}`.

Матрицу со столбцами любого типа и регулируемым расстоянием между ними верстает окружение array, являющееся полным аналогом окружения tabular. Оно имеет аргумент, описывающий последовательность столбцов и разделителей. Его поля набираются в стиле `\textstyle`.

По умолчанию блок array центрируется относительно осевой линии строки математического выражения, но параметр позволяет регулировать его встраивание в формулу. При его значениях b или t нижняя или верхняя строка матрицы совмещается со строкой выражения.

Ячейки матрицы можно объединять командами `\multicolumn` и `\multirow`. При этом, хотя сам блок верстается как формула, колонки «p» и поля, объединенные с помощью команды `\multirow`, формируются как текст.

Столбцы, строки и отдельные ячейки можно отчеркивать, как описано в разд. 3.2. Расстояние между столбцами можно регулировать разделителями `@{•}` (см. разд. 3.2.1).

Как показано в примере, конструкция `\left`• и `\right`• позволяет охватить матрицу array скобками, а команда `\hdotsfor` — заполняет ее ячейки точками. Пакет delarray [32] коллекции tools предлагает другой способ формирования скобок. Он меняет синтаксис окружения array так, что в скобки можно заключить его аргумент:

`\bebin{array}({...})`,

используя в качестве скобок любые разделители, кроме квадратных скобок. Таким же образом можно заключить в скобки и таблицы, верстаемые окружением tabular.

Конструкции

В принципе, блок array позволяет сформировать математические конструкции практически любой сложности, но описывающий их код может оказаться очень громоздким, кроме того, размер `\textstyle` ограничивает возможности верстки ячеек. Поэтому для набора различных конструкций разработаны особые окружения.

`\begin{gathered}[`*привязка*`]`.

Простейшим блоком является окружение gathered, формирующее набор центрированных строк. Его вводит пакет amsmath, а для верстки строк с левым и правым выравниванием пакет mathtools добавляет

блоки lgathered и rgathered. В контексте описанной ранее аналогии их можно рассматривать как таблицы с одной колонкой и разным типом выравнивания. Строки верстаются в стиле `\displaystyle`.

Наборы выровненных строк: gathered , lgathered , rgathered

```
\begin{gathered}[t]
    x+y = a+c, \\
    z = x+d.
\end{gathered}
```

$$x+y=a+c,$$
$$z=x+d.$$

Окружения gathered, lgathered и rgathered имеют параметр, регулирующий встраивание сверстанного выражения в строку формулы аналогично блоку array. По умолчанию оно центрируется относительно осевой линии строки.

`\begin{multlined}[`*привязка*`][`*ширина*`]` .

Для верстки длинных выражений пакет mathtools вводит блок multlined, в котором первая строка прижимается влево, последняя — вправо, а остальные центрируются. Строки формируются в стиле `\displaystyle`. Окружение multlined имеет два параметра. Первый, аналогично окружению array, регулирует встраивание сверстанного выражения в формулу, второй позволяет задать ширину блока, если не устраивает ширина, вычисленная автоматически.

multlined — длинная строка, разбитая на части

```
\begin{multlined}
  y = 2y+x+z-(y+z)+ \\
    -(y+z) +(y+x) -(x+z)+\\
    +(y+x).
\end{multlined}
```

$$\begin{aligned}y &= 2y+x+z-(y+z)+\\ &\quad +(y+x)-(x+z)+\\ &\quad +(y+x).\end{aligned}$$

Строки блока multlined, кроме первой и последней, можно сдвинуть влево или вправо командами

`\shoveleft[`*длина сдвига*`]{`*строка*`}` `\\`,
`\shoveright[`*длина сдвига*`]{`*строка*`}` `\\`.

Для этого сдвигаемую строку нужно целиком, за исключением команды перевода строки, поместить в аргумент одной из команд. По умолчанию команда `\shoveleft` прижимает строку к левой границе блока,

а `\shoveright` — к правой границе. Параметры команд позволяют смещать строки на нужную длину.

split — блок с общей вертикалью выравнивания

```
\begin{split}
    x &= y+z, \\
  y-z &= a+b+z^2.
\end{split}
```

$$\begin{split} x &= y+z, \\ y-z &= a+b+z^2. \end{split}$$

Блок split, вводимый пакетом amsmath, позволяет расщепить длинные строки, или составить систему уравнений. Он имеет одну вертикаль выравнивания, к которой слева и справа прижимаются поля выражений, составляющих строки. Поля набираются в стиле `\displaystyle`.

Блоки с выравниванием и объединяющей скобкой:
cases , cases* , dcases , dcases* , rcases , rcases* , drcases , drcases*

```
x = \begin{cases}
        y,      & y\leq 0,\\
      (y/a)^2, & y\geq 0.
    \end{cases}
```

$$x = \begin{cases} y, & y\leq 0,\\ (y/a)^2, & y\geq 0. \end{cases}$$

Блок cases пакета amsmath, предназначенный для записи вариантов соотношений, объединенных фигурной скобкой, состоит из двух столбцов с левым выравниванием. Первый, прижатый к скобке, предназначен для набора выражений, а второй — для записи условий. Столбцы автоматически разделяются небольшим пустым промежутком. При встраивании в формулу блок центрируется относительно осевой линии ее строки.

Блок cases верстается в стиле `\textstyle`, поэтому большие операторы и дроби смотрятся в нем не очень хорошо. Чтобы устранить это, пакет mathtools вводит блок dcases, верстаемый в размере `\displaystyle`. Вдобавок он определяет блоки rcases и drcases с правыми объединяющими скобками и определяет варианты окружений с именами со звездочкой на конце, в которых поле условия верстается как текст. Математику в этом тексте нужно заключать в строчные формулы:

$$\nabla F = \begin{cases} \dfrac{\partial F}{\partial x}\mathbf{i}_x + \dfrac{\partial F}{\partial y}\mathbf{i}_y + \dfrac{\partial F}{\partial z}\mathbf{i}_z, & \text{декартовы координаты } (x,y,z),\\ \dfrac{\partial F}{\partial \rho}\mathbf{i}_\rho + \dfrac{\partial F}{\partial \phi}\mathbf{i}_\phi + \dfrac{\partial F}{\partial z}\mathbf{i}_z, & \text{цилиндрические координаты } (\rho,\phi,z). \end{cases}$$

```
\nabla F = \begin{dcases*}
       \frac{\partial F}{\partial x} \mathbf{i}_x
      +\frac{•}{•} \mathbf{i}_y  +\frac{\•}{•} \mathbf{i}_z,
   & декартовы координаты $(x,y,z)$,   \\
       \frac{\partial F}{\partial\rho} \mathbf{i}_\rho
      +\frac{•}{•} \mathbf{i}_\phi +\frac{•}{•} \mathbf{i}_z,
   & цилиндрические координаты $(\rho,\phi,z)$.
\end{dcases*}
```

aligned — блок с произвольным числом вертикалей выравнивания

```
\begin{aligned}[b]
      x & = y+z &   y & = a+z  & ...• & •... \\
    a+b & = y   & c+d & = z    & ...• & •...
\end{aligned}
```

$$\begin{aligned}[b]
x &= y+z & y &= a+z & &\dots\bullet\,\bullet\dots\\
a+b &= y & c+d &= z & &\dots\bullet\,\bullet\dots
\end{aligned}$$

`\begin{aligned}[`*привязка*`]` .

Окружение aligned, определяемое пакетом amsmath, имеет параметр, регулирующий встраивание сверстанного блока в формулу аналогично окружению array. По умолчанию блок центрируется относительно осевой линии строки.

В строках блока aligned можно сформировать по нескольку выражений, разделенных между собой пустыми промежутками. Каждое выражение имеет левое и правое поля, которые прижимаются к общей вертикали выравнивания, связывающей выражения, стоящие в разных строках, как показано в примере. Поля, разделяемые знаками &, определены циклически: поле с правым выравниванием первого выражения, затем его поле с левым выравниванием, потом следует пустой промежуток и поле с правым выравниванием следующего выражения и т. д. Выражения верстаются в стиле `\displaystyle`, их количество не ограничено.

Поля, находящиеся в разных строках, формируют столбцы, ширина которых автоматически меняется в зависимости от длин выражений. Между соседними столбцами, содержащими части разных выражений, автоматически вставляются небольшие промежутки.[5] Команды

[5] Точнее говоря, промежуток вставляется между самыми длинными выражениями соседних столбцов. При этом выражения могут находиться в разных строках.

пробелов, находящиеся в начале или в конце этих промежутков, игнорируются, поэтому их ширина не регулируется. Компилятор сам устанавливает ее, исходя из количества и длин выражений, составляющих блок aligned.

Нечетные знаки & создают поля с левым выравниванием, а четные выполняют несколько действий: вставляют промежутки и создают вертикали выравнивания и примыкающие к ним поля с правым выравниванием.

---

alignedat — блок с заданным числом вертикалей выравнивания

```
\begin{alignedat}[b]{3}
    x   & = y+z  & \qquad c+d & = a+z  & ...• & •... \\
                                  .....
\end{alignedat}
```

$$\begin{alignedat}{3} x &= y+z & c+d &= a+z & ...\bullet\bullet... \\ a+b &= y & y &= z & ...\bullet\bullet... \end{alignedat}$$

Блок alignedat пакета mathtools аналогичен блоку aligned, но в нем добавлена возможность регулировать ширину промежутков между выражениями. Его поля формируются по тому же принципу: нечетные знаки & создают поля с левым выравниванием, а четные — с правым. По умолчанию промежутки между выражениями отсутствуют, их нужно задавать вручную, поставив непосредственно после четных знаков & команды пробела, как показано в примере. Их можно поместить в каждой строке, однако для определенности лучше поставить один пробел *перед* самым длинным выражением столбца, а остальные опустить.

Окружение alignedat имеет параметр и аргумент

`\begin{alignedat}[`*привязка*`]{`*количество выражений*`}` .

Аргумент задает количество находящихся в строке выражений. При его значении, равном n, строка может содержать $2n-1$ знака &, из которых $n-1$ будет использовано для разделения выражений и создания их левых полей,[6] и n — для формирования правых полей.

По умолчанию блок alignedat центрируется относительно осевой лини строки формулы, однако параметры t и b позволяют изменять *привязку*, как это обсуждалось ранее.

[6] Напомним, что первое левое поле создается автоматически.

Среди перечисленных блоков aligned обладает наиболее универсальным устройством. Аналогично блокам lgathered и rgathered он позволяет формировать выражения, выровненные по левому и правому краю:

```
\begin{aligned}
  & x+y=1,   \\
  & x+y+z=0,\\
  & x+3y-2z=5.
\end{aligned}
```

$$\begin{aligned} & x+y=1, \\ & x+y+z=0, \\ & x+3y-2z=5. \end{aligned}$$

```
\begin{aligned}
  x+y=1,   \\
  x+y+z=0,\\
  x+3y-2z=5,
\end{aligned}
```

$$\begin{aligned} x+y&=1, \\ x+y+z&=0, \\ x+3y-2z&=5. \end{aligned}$$

Он с успехом заменяет блок split:

```
\begin{aligned}
  x & = a +b +c +d +\\
    & + (y+z)^2.
\end{aligned}
```

$$\begin{aligned} x &= a+b+c+d+ \\ &\quad +(y+z)^2. \end{aligned}$$

С его помощью можно создать аналог блока drcases с альтернативным выравниванием поля условий. Сравните, например:

```
\begin{drcases}
    y\leq0, & y,\\
    y\geq0, a>0, &\sqrt{y/a}.
\end{drcases} = x.
```

$$\begin{drcases} y\leq0, & y \\ y\geq0,\, a>0, & \sqrt{y/a}. \end{drcases} = x.$$

```
\left.
  \begin{aligned}
    y\leq0, && y\\
    y\geq0, a>0, && \sqrt{y/a}
  \end{aligned} \,
\right \} = x.
```

$$\left.\begin{aligned} y\leq0, && y \\ y\geq0,\, a>0, && \sqrt{y/a} \end{aligned}\right\} = x.$$

4.4. Нумерованные формулы

4.4.1. Нумерация формул

Для верстки вынесенных формул используются окружения, перечисленные в табл. 4.10. Все они, кроме equation и eqnarray, определены пакетом amsmath.

Таблица 4.10

Математические окружения

equation	multline	align	flalign
eqnarray	gather	alignat	xalignat

Созданные с их помощью формулы автоматически нумеруются. Для ссылок на них определены команда `\eqref{`*метка*`}`, выводящая номер формулы, заключенный в круглые скобки, и команда `\ref{`*метка*`}`, печатающая номер без скобок. Чтобы сослаться на формулу, ее нужно пометить командой `\label{`*метка*`}`. Если окружение содержит одну формулу, метку можно поставить как перед ней, так и после нее. В окружениях, содержащих нескольких формул, все нужно пометить командами `\label`, поставив их в начале или конце формул до команды `\\`, разделяющей формулы. В последней строке команду `\\` использовать не стоит, так как она создаст пустую строку с номером.

Поставив вместо метки команду `\nonumber` или `\notag`, можно отменить нумерацию *отдельной* формулы, а чтобы подавить нумерацию *всех* формул, в имя окружения нужно добавить звездочку. Ненумерованные варианты имеют все перечисленные в табл. 4.10 окружения.

Окружению, содержащему несколько нумерованных формул, собственный номер не присваивается. Это видно из следующего примера:

```
\begin{eqnarray}\label{e:array}
   x &=& y+a \label{e:x}\\
   z &=& a+x \label{e:z}
\end{eqnarray}
```

$$x = y + a \tag{4.1}$$
$$z = a + x \tag{4.2}$$

Сошлемся на «систему» `\eqref{e:array}` ↝ (4.1) и оба ее уравнения: `\eqref{e:x}` ↝ (4.1), `\eqref{e:z}` ↝ (4.2). Из напечатанных номеров видно, что метки `e:sample` и `e:x` ассоциированы с первым уравнением, так как обе они находятся в одной строке-формуле.

Чтобы присвоить номер отдельным уравнениям и системе в целом, используйте окружение subequations:

```
\begin{subequations}\label{e:array}
   \begin{eqnarray}
      x &=& y+a \label{e:x}\\
      z &=& a+x \label{e:z}
   \end{eqnarray}
\end{subequations}
```

$$x = y + a \tag{4.3a}$$
$$z = a + x \tag{4.3b}$$

В него помещается одно или несколько окружений с формулами и метками уравнений, а также метка общего номера системы. Получив текущее значение счетчика уравнений, окружение subequations присваивает его всем формулам и дополнительно помечает их латинскими буквами. Например, в систему `\eqref{e:array}` ↝ (4.3) входят уравнения `\eqref{e:x}` ↝ (4.3a) и `\eqref{e:z}` ↝ (4.3b).

4.4.2. Структура формул

Большинство конструкций, применяемых для верстки нумерованных формул, аналогичны описанными ранее математическим блокам.

equation — нумерованная формула

```
\begin{equation}\label{e:...}
    r =  sqrt{x^2 +y^2 +z^2}.
\end{equation}
```

$$r = \sqrt{x^2 + y^2 + z^2}. \quad (4.4)$$

Окружения equation и multline верстают формулы, которым присваивается только один номер, хотя содержащиеся в них выражения могут состоять из нескольких строк. В multline возможность формирования многострочных выражений заложена в самом окружении, а в случае equation для этого нужно использовать блоки.

multline — формула с продолжением

```
\begin{multline}
  y = 2y+x+z- \\
    -(y+z)-(y+x)+\\
    +(y+x).
  \label{e:...}
\end{multline}
```

$$\begin{aligned} &|\leftarrow \\ &y = 2y + x + z- \\ &\quad - (y + z) - (y + x)+ \\ &\qquad\quad \triangleright|\triangleleft \qquad -(y+x). \\ &\qquad\qquad\qquad\quad \rightarrow| \end{aligned} \quad (4.5)$$

Окружение multline, являющееся аналогом блока multlined, предназначено для верстки длинных выражений. Оно может содержать несколько строк, первая из которых прижимается влево, последняя — вправо, а остальные центрируются. Номер формулы проставляется в конце последней строки.

Команды `\shoveleft{`*строка*`}` `\\` и `\shoveright{`*строка*`}` `\\` дают возможность изменить выравнивание средних строк. Первая выравнивает их по началу первой строки, а вторая по концу последней. Для этого выравниваемая *строка* должна быть полностью, за исключением команды `\\`, помещена в аргумент одной из команд.

В отличие от блока multlined, окружение multline не имеет параметра, регулирующего ширину верстаемой формулы, а команды `\shoveleft` и `\shoveright` не имеют параметра, позволяющего сдвигать строки на требуемую длину.

gather — набор центрированных формул

```
\begin{gather}\label{e:...}
    x+y = a+c,  \\
    z = x+d.    \label{e:...}
\end{gather}
```

$$x + y = a + c, \tag{4.6}$$
$$z = x + d. \tag{4.7}$$
$$\triangleright|\triangleleft$$

Окружение gather, аналогом которого является блок gathered, предназначено для набора нескольких центрированных формул.

Структура окружений, обсуждаемых далее, более сложна. Их формулы формально представляют собой строки таблицы с выровненные полями (ячейками), разделенными знаком &. Одинаковые поля разных формул (строк) образуют столбцы, связанные общей вертикалью выравнивания.

eqnarray — набор уравнений с тремя полями:
«выравнивание справа – центрирование – выравнивание слева»

```
\begin{eqnarray} \label{e:...}
  x &=& a+z, \\
  xy &=& a+b+z^2. \label{e:...}
\end{eqnarray}
```

$$\begin{array}{rcl} \multicolumn{3}{c}{\&} \\ \rightarrow| & \triangleright|\triangleleft & |\leftarrow \\ x & = & a+z, \\ xy & = & a+b+z^2. \end{array} \tag{4.8}$$

(4.9)

Окружение eqnarray служит для записи систем уравнений. В нем предусмотрено три поля: знаки соотношений составляют центрированный столбец, к которому прижимаются левые и правые части уравнений, образуя таким образом столбцы с правым и левым выравниванием. Части уравнений верстаются в стиле `\displaystyle`, а поле соотношений — в стиле `\textstyle`. Для разделения полей конструкции eqnarray достаточно двух знаков &, поэтому попытка добавить еще один приведет к ошибке.

Окружение eqnarray, считающееся устаревшим, поддерживается, главным образом, из-за большого количества «старых» документов, содержащих верстаемые им формулы. В настоящее время вместо него рекомендуется использовать конструкцию align, аналогичную блоку aligned. Она обладает более широкими возможностями и верстает более компактные формулы. Иногда «отсутствие компактности» может оказаться более выигрышным, поэтому окружение eqnarray все же находит свое применение. Один из примеров показан далее, а здесь для сравнения приведем простое выражение, сверстанное двумя способами:

```
\begin{eqnarray*}
    x+y+z   & = & 0,\\
    x+3y-2z & = & 5.
\end{eqnarray*}
```

$$\begin{aligned} x+y+z &= 0,\\ x+3y-2z &= 5. \end{aligned}$$

```
\begin{align*}
    x+y+z   & = 0,\\
    x+3y-2z & = 5.
\end{align*}
```

$$\begin{aligned} x+y+z &= 0,\\ x+3y-2z &= 5. \end{aligned}$$

---

align — набор уравнений с последовательностью полей «правое – левое – вычисляемый промежуток»...

```
\begin{align} \label{e:...}
        x & = y+z & y & =  a+z & ...• & •...,
      a+b & = y   & c+d & = z  & ...• & •...
\end{align}
```

$$\begin{aligned} x &= y+z & y &= a+z & &\ldots\bullet\,\bullet\ldots, & (4.10)\\ a+b &= y & c+d &= z & &\ldots\bullet\,\bullet\ldots & (4.11) \end{aligned}$$

Как уже говорилось, устройство окружения align аналогично блоку aligned. Поля в нем определены циклически: два соседних столбца с правым и левым выравниванием связываются общей вертикалью и т. д. Количество полей (столбцов) не ограничено. Соседние выражения, не связанные общей вертикалью, автоматически разделяются промежутком (см. примечание на с. 110).

Ранее на примере блока aligned мы рассмотрели различные варианты выравнивания многострочных выражений. Сказанное напрямую относится и к конструкциям, формируемым окружением align.

---

flalign — набор уравнений с последовательностью полей «правое – левое – максимальный промежуток»...

```
\begin{flalign} \label{e:...}
        x & = y+z &   y & = a+z & ...• & •...,
      a+b & = y   & c+d & = z   & ...• & •...
\end{flalign}
```

$$\begin{aligned} x &= y+z & y &= a+z & &\ldots\bullet\,\bullet\ldots, & (4.12)\\ a+b &= y & c+d &= z & &\ldots\bullet\,\bullet\ldots & (4.13) \end{aligned}$$

В окружении align все выражения, в том числе и крайние, окружены слева и справа одинаковыми свободными полями, это приводит к центрированию уравнений на странице. В окружении flalign ширина полей между выражениями делается максимальной, при этом крайние выражения прижимаются к краю страницы и номеру формулы.

---

alignat — набор уравнений с последовательностью полей «правое – левое – заданный промежуток»...

```
\begin{alignat}{3} \label{e:...}
      x & = y+z & \qquad c+d & = a+z & ...• & •...,
    a+b & = y   &          y & = z   & ...• & •...
\end{alignat}
```

$$\underset{\rightarrow\|\leftarrow}{\&} \quad \underset{|\longleftrightarrow|}{\&} \quad \underset{\rightarrow\|\leftarrow}{\&} \quad \underset{\rightarrow\|\leftarrow}{\&}$$

$$x = y + z \qquad c + d = a + z ... \bullet\bullet ..., \tag{4.14}$$

$$a + b = y \qquad y = z \qquad ... \bullet\bullet ... \tag{4.15}$$

Если в окружениях align и flalign ширина пустых полей между выражениями вычисляется автоматически, то в окружении alignat ее устанавливает автор. Данное окружение имеет аргумент, указывающий количество вертикалей выравнивания. При его значении, равном n, строка может содержать n выражений и $n-1$ промежуток между ними. Ширина промежутков по умолчанию равна нулю. Чтобы задать ее, после четных знаков & нужно поставить команду-пробел требуемой ширины, как обсуждалось на с. 111 для блока alignedat.

---

xalignat — набор уравнений с последовательностью полей «правое – левое – вычисляемый + заданный промежуток»...

```
\begin{xalignat}{3} \label{e:...}
       x & = y+z & \qquad c+d & = a+z & ...• & •...,
     a+b & = y   &          y & = z   & ...• & •...
\end{xalignat}
```

$$\underset{\rightarrow\|\leftarrow}{\&} \quad \underset{\longmapsto|\longleftrightarrow|}{\&} \quad \underset{\rightarrow\|\leftarrow|}{\&} \quad \underset{\longmapsto|}{\&} \quad \underset{\rightarrow\|\leftarrow}{\&}$$

$$x = y + z \qquad c + d = a + z \qquad ... \bullet\bullet ..., \tag{4.16}$$

$$a + b = y \qquad y = z \qquad ... \bullet\bullet ... \tag{4.17}$$

Окружение xalignat позволяет регулировать ширину промежутка между выражениями, заботясь при этом, чтобы она не оказалась нулевой. Длина пробела, заданного вручную, добавляется к ширине, вычисленной автоматически. Окружения xalignat и alignat имеют одинаковый синтаксис.

4.5. Разбивка формул

Формулы, находящиеся в строке, разбиваются как обычные строки. Точками автоматического разрыва служат знаки сравнения или присваивания и операторы, а для принудительного разрыва можно пользоваться командами `\\[•]` и `\linebreak[.]`

Формулы, создаваемые окружениями equation и multline, размещаются на странице целиком, даже если их выражения состоят из нескольких строк. Наборы из нескольких уравнений,[7] верстаемых одним окружением, как правило, тоже размещаются на одной странице. Среди формирующих их окружений только eqnarray автоматически переносит уравнения, не умещающиеся на текущей странице, на следующую. В остальных случаях для этого требуется вмешательство автора. С этой целью пакет amsmath вводит команду `\displaybreak[`*число*`]`, которую нужно поставить перед командой перевода строки \\, чтобы перенести часть уравнений на следующую страницу. Целое *число* в ее параметре, принимающее значения от нуля до четырех, регулирует «жесткость» указания компилятору от рекомендации (0), до используемого по умолчанию обязательного выполнения (4). Добавив в преамбулу декларацию `\allowdisplaybreak[`*число*`]`, можно разрешить автоматическое разбиение уравнений во всем документе. Ее параметр регулирует действия компилятора аналогично параметру команды `\displaybreak`. Когда разрешен автоматический перенос уравнений, команда `\\*[•]` позволяет «склеить» строки, которые нужно напечатать вместе, и отрегулировать расстояние между ними.

Часто формулы нужно связать кратким комментарием или логической связкой. Это можно сделать непосредственно в верстающем их окружении. Команда `\intertext{`*текст*`}` автоматически разрывает строки уравнений и вставляет между ними *текст*, находящийся в ее аргументе:

[7] Напомним, что уравнениями считаются лишь те выражения, что потенциально могут иметь собственные номера.

```
\begin{gather}
   x \equiv z,
\intertext{если}
   x=y,
\shortintertext{а также }
   y=z.
\end{gather}
```

$$x \equiv z, \tag{4.18}$$

если

$$x = y, \tag{4.19}$$

а также

$$y = z. \tag{4.20}$$

Вставляемый текст смотрится хорошо, если он занимает несколько строк, или почти полную строку. Отбивка коротких фраз зачастую оказывается слишком велика, поэтому пакет mathtools вводит альтернативную команду `\shortintertext{текст}` с меньшей отбивкой. Аналогично своему прототипу она тоже автоматически разрывает уравнения. Действие обеих команд показано в предыдущем примере.

4.6. Верстка сложных формул

Сложные формулы собираются подобно конструктору. Вначале выбирается оптимальное окружение (таблица), связывающее строки формул, затем, при необходимости, поля выражений (ячейки таблицы) усложняются с помощью блоков. На конечном этапе отдельные части настраиваются так, чтобы вся конструкция имела оптимальный вид. Поясним сказанное, разобрав код следующего примера:

$$I + \delta I = \begin{cases} I_c[2f(r,b) - f(r,a)], & r \le a, \\ I_c[2f(r,b) - 1], & a \le r \le b, \\ I_c, & b \le r \le R, \end{cases} \tag{4.21}$$

$$\frac{B_z + \delta B_z}{\mu_0} = \begin{cases} H + I_c \left[P(r,a) - \dfrac{1}{2\,\mathrm{ch}(R/r)} \right], & a \le r \le b, \\ H - \delta H + I_c \left[\dfrac{1}{2\,\mathrm{ch}(R/r)} + \right. & \\ \qquad \left. + P(r,a) - 2P(r,b) \right], & B \le r \le R. \end{cases} \tag{4.22}$$

Основной конструкцией, связывающей уравнения для тока I и индукции поля B_z, является окружение eqnarray, формирующее формулы (4.21) и (4.22). В данном случае его выбор вполне оправдан, так как в громоздких выражениях eqnarray лучше выделяет структуру формул:

```
\begin{eqnarray}\label{e:I-hysteresis}
  I +\delta I
    &=& \begin{dcases}
          I_c[ 2f(r,b) -f(r,a) ], & r \leq a\\
          I_c[ 2f(r,b) -1 ],      & a \leq r \leq b,\\
          I_c,
           \hphantom{H+I_c\left[P(r,a)-\frac{1}{2\ch(R/r)}\right]}
                                   & b \leq r \leq R,
        \end{dcases}\\
  \frac{B_z +\delta B_z}{\mu_0}
    &=& \begin{dcases}
          H +I_c \left[ P(r,a) -\frac{1}{2\ch(R/r)} \right],
              \hphantom{I_c}
                                   & a \leq r \leq b,\\
          \begin{split}
            H -\delta H & +I_c \biggl[ \frac{1}{2\ch(R/r)} +\\[-1ex]
                        & +P(r,a) -2P(r,b) \biggr],
          \end{split}
                                   & B \leq r \leq R.
        \end{dcases}
  \label{e:Bz-hysteresis}
\end{eqnarray}
```

Набор условий, определяющих гистерезис тока, оформлен с помощью блоков dcases, выбранных для оптимизации печати дробей второй формулы. Блоки, находящиеся в разных строках окружения eqnarray, между собой не связаны, однако столбцы их условий желательно выровнить одинаково, сделав одинаковой ширину столбцов их значений. Для этого в первом окружении к выражению, имеющему минимальную длину, добавлен фантом самого длинного выражения из второго окружения, а во втором окружении к нему добавлен фантом короткого выражения. Можно поступить и проще, поставив в конце короткого выражения пробел подобранной длины, однако фантомы выровнят столбцы автоматически.

Длина второго выражения во втором блоке слишком велика, поэтому оно разбито блоком split. Выражение содержит дробь, определяющую размер квадратных скобок, заданный командами `\biggl` и `\biggr`. Так как дробь стоит только в одной строке выражения, такой метод регулировки размеров более удобен, чем конструкция `\left`•...`\right`•. Высокие скобки излишне раздвигают строки обсуждаемого выражения, поэтому интервал между ними уменьшен с помощью команды перевода строки `\\[-1ex]`.

4.7. Теоремы

Математика — наука строгая, требующая доказательства выдвинутых утверждений. Их формы могут быть очень разнообразны, например аксиома, теорема, лемма, следствие и т. д. Некоторые из них, как аксиома, не доказываются. Доказательство других может быть очень коротким, или, напротив, занять несколько страниц. В ходе доказательства могут возникать новые утверждения, которые тоже нужно доказывать. Возникающие таким образом иерархические структуры утверждений и доказательств невозможно свести к какой-то единой, наперед заданной форме, поэтому LaTeX дает возможность строить их самостоятельно. Автор сам должен выработать формы утверждений и их подчиненность, при необходимости создавая для этого новые окружения.

Будем использовать термин теорема как синоним всех форм утверждений и опишем средства их формирования, предоставляемые ядром LaTeX и стандартным пакетом amsthm [33], который нужно добавить в список загружаемых пакетов, в документах, содержащих теоремы.

Простейшую форму теорем вводит команда:

\newtheorem*{имя}{форма утверждения}.

Она создает окружение, имя которого должно быть уникальным. Ее второй аргумент задает заголовок, выводимый перед утверждением, формируемым этим окружением. Поясним сказанное примером:

```
\newtheorem*{theorem}{Теорема}
   ...
\begin{theorem}[Ферма]
  ...
\end{theorem}
```

Теорема (Ферма)**.** *Уравнение* $m^n + l^n = k^n$ *не имеет решения для целых чисел* $n > 2$, m, k, l.

Создаваемое окружение имеет параметр, позволяющий уточнить форму утверждения. Чаще всего в нем, как в приведенном примере, указывается автор теоремы. Теоремы, определяемые командой \newtheorem* не нумеруются.

Для оформления доказательств предназначено окружение proof:

\begin{proof}[заголовок].

```
\begin{theorem}
  Волга впадает в Каспийское море.
\end{theorem}
```

Теорема. *Волга впадает в Каспийское море.*

```
\begin{proof}
  Смотри географическую карту.
\end{proof}
```

Доказательство. Смотри географическую карту. □

По умолчанию, в начале доказательства выводится заголовок *Доказательство*, а в его конце автоматически ставится знак □, означающий, «что и требовалось доказать». Параметр окружения proof позволяет заменить стандартный заголовок любым другим, или изменить его шрифт. Команда `\qedhere` печатает знак □ в любом месте доказательства внутри окружения proof, отжимая его направо.[8] Чтобы напечатать его вне окружения, используйте команду `\qed`. Изменив команду `\qedsymbol` (см. разд. 7.1), знак □ можно заменить другим символом, `\renewcommand\qedsymbol\ast ... \qed` ⇝ ∗, или отменить его вывод, сделав символ пустым: `\renewcommand\qedsymbol{}`.

Окружение proof используется для простых доказательств, не требующих дополнительных утверждений Длинные доказательства со сложной структурой можно разбить на части командами разделов `\section`, `\subsection` и т. д.

Наиболее общую форму теорем вводит команда:

`\newtheorem{`*имя*`}[`*счетчик*`]{`*форма утверждения*`}`.

Она определяет новое окружение и ассоциирует с ним счетчик, получающий такое же имя. Теоремам присваиваются номера, а метки, поставленные внутри окружений, позволяют сослаться на них. Параметр команды `\newtheorem` дает возможность использовать существующий счетчик для нумерации некоторой группы утверждений.[9] Это позволяет вести общий подсчет нескольких форм утверждений, например постулатов и аксиом:

```
\newtheorem{axiom}{Аксиома}
\newtheorem{postulate}[axiom]{Постулат}
    ...
\begin{postulate}[для атеистов]
  Бога нет!    \label{•}
\end{postulate}
\begin{axiom}[для верующих]
  Бог есть!    \label{•}
\end{axiom}
```

Постулат 1 (для атеистов)**.** *Бога нет!*

Аксиома 2 (для верующих)**.** *Бог есть!*

[8] Если команда `\qedhere` дает ошибку, поместите ее в бокс: `\mbox{\qedhere}`.

[9] При этом счетчик с именем окружения тоже вводится и отождествляется с существующим счетчиком.

По умолчанию заголовок утверждения и его уточнение разделяются номером, декларация `\swapnumbers` позволяет напечатать номер в начале. Ее нужно использовать до определения теорем. Например:

```
\swapnumbers
\newtheorem{aphorism}{Афоризм}
    .....
\begin{aphorism}[Козьма...]
    .....
\end{aphorism}
```

1 Афоризм (Козьма Прутков)**.** *Нельзя объять необъятное.*

Нумерацию утверждений, возникающих в ходе доказательства основной теоремы, можно согласовать с ее номером. Для этого имеется еще одна форма определения теорем:

`\newtheorem{`*имя*`}{`*форма утверждения*`}[`*счетчик*`]` .

В параметре данной команды указывается имя основной теоремы, номер которой выводится перед номером утверждения. При формировании новой теоремы такие утверждения подсчитываются заново.

Эту форму определения можно использовать, чтобы нумеровать теоремы по главам или разделам. Тогда в параметре следует указать счетчик раздела, значение которого будет выводиться перед номером теоремы. Проиллюстрируем сказанное примерами:

```
\swapnumbers
\newtheorem{mat}{Теорема}
\newtheorem{lemma}{Лемма}[mat]
    ...
\begin{mat}[материализм]
  Все родилось из ничего.
\end{mat}
    ...
\begin{lemma}
  Материя существует вечно.
\end{lemma}
    ...
\begin{lemma}
  Вечность --- это...
\end{lemma}
```

1 Теорема (материализм)**.** *Все родилось из ничего.*
...

1.1 Лемма. *Материя существует вечно.*
...

1.2 Лемма. *Вечность — это движение времени по замкнутой траектории.*
...

Отметим, что обе формы определения нумерованных теорем имеют вариант «со звездочкой» `\newtheorem*{•}{•}`, обсуждавшийся первым, который создает теоремы без номеров.

По умолчанию все утверждения оформляются в одном стиле. Воспользовавшись декларацией `\theoremstyle{`*стиль*`}` , можно выбрать

один их трех предустановленных стилей, конкретная реализация которых зависит от класса документа. Это нужно сделать до определения окружений командами `\newtheorem`.

В стиле plain, используемом по умолчанию в стандартных классах, заголовок теоремы делается жирным, а ее текст набирается курсивом, как показано в примерах выше. В стиле definition заголовок также жирный, но для текста используется обычный шрифт. В стиле remark для заголовка выбран курсив, а для текста обычный шрифт:

```
\theoremstyle{definition}
\newtheorem*{def}{Определение}
    ...
```

Определение. Интернет это виртуальная реальность, данная нам в искушение.

```
\theoremstyle{remark}
\newtheorem*{rem}{Замечание}
    ...
```

Замечание. E pur si muove! (Galileo Galilei)

Стандартный пакет theorem [34] коллекции tools вводит дополнительные стили оформления теорем и позволяет настроить шрифты заголовков и формулировок, а также их вертикальные отбивки.

Подводя итог, сделаем важное замечание. Декларации `\newtheorem` являются глобальными, область их действия не ограничивается группой. Определения теорем следует размещать в преамбуле. Приведем пример определения списка теорем:

```
\swapnumbers
\newtheorem{axiom}{Аксиома}
\newtheorem{theorem}{Теорема}
\newtheorem{lemma}{Лемма}[theorem]

\theoremstyle{definitin}
\newtheorem*{definition}{Определение}

\theoremstyle{remark}
\newtheorem{sequence}{Следствие}
    ....
\begin{document}
```

Для теорем, следствий и аксиом мы выбрали независимую нумерацию, номера лемм подчинены номеру теоремы, а определения не нумеруются. Номера утверждений будут печататься перед заголовками. Заголовки теорем, лемм и аксиом имеют стандартное шрифтовое оформление. Определения верстаются в стиле definition, а следствия — в стиле remark.

4.8. Коммутативные диаграммы

В математике, в частности в теории категорий, коммутативная диаграмма связывает объекты определенной категории путями — *морфизмами*. Для наглядности она изображается виде структуры наподобие графа, вершинами которого служат объекты, а ребрами — морфизмы.

Для построения коммутативных диаграмм пакет amscd [35] коллекции $\mathcal{AMS}$-LaTeX вводит окружение CD, которое является математическим блоком с особым синтаксисом. Диаграмма представляет собой прямоугольную сетку с узлами, связанными стрелками или знаками равенства. В узлах, помеченных в приведенном примере буквами, могут находиться любые выражения:

$$\begin{array}{ccccc} A & \xleftarrow{a} & B & \xrightarrow[b]{} & C & = & D \\ \| & & \uparrow{\scriptstyle c} & & d\downarrow & & \\ H & & F & \xleftarrow{e} & G & \xrightarrow[f]{} & E \end{array} \tag{4.23}$$

```
\begin{equation}
   \begin{CD}
     A  @<a<<  B   @>>b>  C   @=  D \\
     @|       @AAcA       @VdVV     @.\\
     H  @.     F   @(e((  G  @))f) E
   \end{CD}
\end{equation}
```

Сетка формируется строками двух типов. Одни состоят из узлов и горизонтальных связей, а другие — только из вертикальных связей. Если связь отсутствует, узлы разделяются командой `@.`, необходимой для поддержания баланса узлов и связей.

Морфизмы печатают команды, собранные в табл. 4.11. В ее верхней части представлены горизонтальные связи, а в нижней — вертикальные. В примерах они связывают узлы A и B. Вертикальные связи, перечисленные в первой колонке сверху вниз, в примере показаны слева направо.

Команды имеют специфический синтаксис. Их имена начинаются символом `@`, за которым следует символ, определяющий морфизм. Для вывода знаков горизонтального и вертикального равенства используются символы `=` и `|`. В командах, рисующих стрелки, символы указывают их направление. В обозначениях горизонтальных стрелок

использованы знаки сравнения и круглые скобки, например `@<<<` (←) и `@)))` (→). Открывающая скобка служит «синонимом» знаку меньше, а закрывающая — знаку больше. Команды `@AAA` (↑) и `@VVV` (↓) рисуют вертикальные стрелки.

Таблица 4.11

Морфизмы коммутативных диаграмм

Команды	Примеры
`@.`	AB
`@=`	$A = B$
`@>{вверху}>{внизу}>` `@){вверху}){внизу})`	$A \xrightarrow[\text{внизу}]{\text{вверху}} B$
`@<{вверху}<{внизу}<` `@({вверху}({внизу}(`	$A \xleftarrow[\text{внизу}]{\text{вверху}} B$

Команды				
`@.`	A	A	A	A
`@|`		$\|\|$	слева $\uparrow$ справа	слева $\downarrow$ справа
`@A{слева}A{справа}A`				
`@V{слева}V{справа}V`	B	B	B	B

Стрелки могут иметь верхние и нижние или левые и правые индексы, которые ставятся между парой символов, указывающих направление, например: `@>a>c>` ⇝ $\xrightarrow[c]{a}$. Сложные индексы лучше заключить в фигурные скобки: `@>{g\circ h}>>` ⇝ $\xrightarrow{g\circ h}$. Таким же образом следует поступить, если индекс содержит символ, используемый в имени команды: `@>{a>b}>>` ⇝ $\xrightarrow{a>b}$.

Ресурсов окружения CD недостаточно для создания диаграмм с многообразными типами связей и сложной логикой, особенно, если их объекты имеют более четырех связей. Для их верстки лучше использовать пакет xy из коллекции XY-pic, предоставляющий самые широкие возможности. Он позволяет делать дополнительные (диагональные) связи и генерировать любой тип стрелок, включая стрелки с управляемой кривизной и волнистой, пунктирной и точечной линией. Его оригинальные команды и конструкции имеют довольно сложный синтаксис, подробно описанный в документации пакета [36].

4.9. Химические формулы

Химические формулы являются частью текста. Хотя очень часто в них встречаются различные индексы, как обозначения элементов, так и индексы должны печататься текстовыми шрифтами. Совершенно неправильно печатать серную кислоту в виде математической формулы: `$H_2SO_4$` ↝ H_2SO_4. Часто прибегают к математическому набору индексов, но такой набор тоже некорректен, так как при изменении шрифта наименования элементов и индексов будут выглядеть по-разному:

`{\sf H$_2$SO$_4$}` ↝ {\sf H$_2$SO$_4$}.

Для печати индексов в тексте в ядре LaTeX определены команды `\textsubscript{`*текст*`}` и `\textsuperscript{`*текст*`}`. Они напечатают индексы тем же шрифтом, что и элементы:

`{\sf H\textsubscript{2}SO\textsubscript{4}}` ↝ {\sf H\textsubscript{2}SO\textsubscript{4}}.

Аналогично математическим индексам, индексы текста допускают вложение: `2\textsubscript{2\textsubscript{2}}` ↝ 2_{2_2}.

Сказанное выше относится и к обозначениям единиц измерения, например, `A·м\textsuperscript{-2}` ↝ A·м^{-2}. Отметим, что знак умножения можно печатать непосредственно символом (код `U+b7`) или командой `\textperiodcentered`.

Пакет chemformula [37] позволяет сильно упростить набор химических формул. Для этого он вводит команду

`\ch[`*настройки*`]{`*химическая формула*`}`,

имеющую собственный синтаксис, обеспечивающий верстку формул со сложной структурой. Разберем его на примерах, в которых опустим стрелки, связывающие код и сверстанные формулы.

Формулу разбивают на блоки, разделяемые пробелами. В блоках буквы и скобки печатаются как обычный текст, а цифры — нижними индексами:

`\ch{Sb2O3}` ↝ \ch{Sb2O3}, `\ch{Sb 2O3}` ↝ \ch{Sb 2O3}, `\ch{Sb 2 O3}` ↝ \ch{Sb 2 O3}.

Формула первого примера состоит из одного блока. Во втором примере она разбита на два блока, и индекс 2 относится к блоку с кислородом. Третья формула разбита на три блока, двойка образует самостоятельный блок и потому печатается как обычный текст. Из примеров видно, что пробелы между блоками изменяются в зависимости от контекста. Соберем по изложенным правилам более сложную формулу:

`\ch{KCr(SO4)2 * 12 H2O}` ↝ \ch{KCr(SO4)2 * 12 H2O}

Пример демонстрирует, что звездочки (или точки) заменяются знаком умножения в виде центрированной точки.

Верхние индексы в блоках указывают явно, при этом знаки плюс и минус автоматически становятся верхними индексами. Считается, что они относятся к зарядам ионов или комплексов, например

`\ch{[AgCl2]-}` $\rightsquigarrow [AgCl_2]^-$, `\ch{Y^3+}` $\rightsquigarrow Y^{3+}$, `\ch{^{227}90Th}` $\rightsquigarrow {}^{227}_{90}Th$.

Нижние индексы, содержащие символы, отличные от цифр, тоже задают явно:

`\ch{A_nB}` $\rightsquigarrow A_nB$, `\ch{A_{$n$}B}` $\rightsquigarrow A_nB$, `\ch{NO^*}` $\rightsquigarrow NO^*$.

Из последнего примера видно, что в индексах звездочки не преобразуются. По умолчанию индексы печатаются текстовыми шрифтами, но, как показано во втором примере, их можно оформить в виде строчных формул, в которых доступны все математические алфавиты и символы.

Формулы могут содержать «сторонний текст» и математические выражения. Текст заключают в двойные или одинарные кавычки, а выражения в знаки доллара:

```
\ch{"\dots\ " Na + "\dots\ " Cl2 $\ \to\ $ "\dots\ " NaCl} ⇝
```

⇝ … Na + … Cl_2 → … NaCl

`\ch{$2n$ Na + $n$ Cl2 $\ \to\ 2n$ NaCl}` ⇝ $2n\,Na + n\,Cl_2 \to 2n\,NaCl$.

Обычные пробелы игнорируются как в тексте, так и в выражениях, поэтому их нужно задавать командами, однако команды `\hspace` с этой целью использовать нельзя, так как их аргумент содержит описания длин, нарушающие синтаксис команд `\ch`. В то же время в формулах прекрасно работают пружины. В самой формуле пробелами разделяют блоки, а чтобы «напечатать» пробел, следует добавить в блок символ «~», например `\ch{B A~B}` BA B.

Стехиометрические коэффициенты оформляют в виде отдельных блоков. Коэффициенты, не являющиеся целыми числами, преобразуются следующим образом:

`\ch{ .5 A }` $\rightsquigarrow 0.5\,A$, `\ch{ 1/2 A }` $\rightsquigarrow \frac{1}{2}\,A$,

`\ch{ (.5) B }` $\rightsquigarrow (.5)\,B$, `\ch{(1/2) B}` $\rightsquigarrow (1/2)\,B$,

`\ch{(.5)1/2 C}` $\rightsquigarrow \frac{(.5)1}{2}\,C$, `\ch{1_1/2 C}` $\rightsquigarrow 1\frac{1}{2}\,C$.

Слэш формирует дробь. К действительным числам, у которых отсутствует целая часть, добавляется цифра ноль. Числа, помещенные в круглые скобки, не преобразуются. Символ «_» сшивает числа.

Команды `\bond{`*тип связи*`}` формируют химические связи. Они могут входить в состав блоков или стоять отдельно. Для ввода наиболее

часто встречающихся связей используют символы `+`, `-` и `=`, которые должны связывать части одного блока:

`\ch{ A-B=C+D \bond{<co} E\bond{co>}F}` ⇝ $\mathrm{A{-}B{=}C{\equiv}D{\leftarrow}E{\rightarrow}F}$.

Типы обозначения связей и приведены в табл. 4.12.

Таблица 4.12

Типы химических связей

Символ	Аргументы	Связь	Аргументы	Связь
—	single, normal, sb	−	semisingle, dotted	⋯
=	double, db	=	semidouble, deloc	⋍
+	triple, tp	≡	semitriple, tdeloc	≣
	coordleft, <co	←	coordright, co>	→

Уточним поведение символов `+` и `-` в формулах. В конце блоков они преобразуются в верхние индексы, а внутри них — в связи. Лишь стоя отдельно они остаются знаками плюс и минус:

`\ch{A3+ ~A+3 ~A+^3 ~A + B}` ⇝ $\mathrm{A_3^+\ A{\equiv}_3\ A{\equiv}^3\ A + B}$

`\ch{A3- ~A-3 ~A-^3 ~A - B}` ⇝ $\mathrm{A_3^-\ A{-}_3\ A{-}^3\ A - B}$.

Блоки, состоящие из символа `^` или буквы `v`, печатают стрелки, обозначающие выпадение осадков или образование газообразных компонент:

`\ch{Ag + 2 HNO3 -> AgNO3 + NO2 ^ + H2O}` ⇝

⇝ $\mathrm{Ag + 2\,HNO_3 \longrightarrow AgNO_3 + NO_2{\uparrow} + H_2O}$,

`\ch{NaCl + AgNO3 -> AgCl v + NaNO3}` ⇝

⇝ $\mathrm{NaCl + AgNO_3 \longrightarrow AgCl{\downarrow} + NaNO_3}$.

Для индикации типов химических реакций определен набор стрелок, печатаемых командой `\charrow{mun}[сверху]{снизу}`. Параметры команды позволяют «подписать» стрелку *сверху* и *снизу*. Текст «подписей» может содержать несколько строк, разбитых командами `\\`. Он верстается шрифтом `\scriptsize` в центрированной колонке окружения tabular.

Каждый *mun* стрелок имеет мнемонический код, заменяющий при наборе команду `\charrow`. Коды представляют собой последовательности символов, набираемых в виде отдельных блоков, например:

`\ch{2 Na + Cl2 -> 2 NaCl}` ⇝ $\mathrm{2\,Na + Cl_2 \longrightarrow 2\,NaCl}$,

`\ch{A <->[a][b] B}` ⇝ $\mathrm{A \underset{b}{\overset{a}{\longleftrightarrow}} B}$.

Мнемонические коды и печатаемые ими стрелки приведены в табл. 4.13.

Таблица 4.13

Обозначения направлений химических реакций[10]

Код	Стрелка	Код	Стрелка	Код	Стрелка
`<-`	$\longleftarrow$	`->`	$\longrightarrow$	`<->`	$\longleftrightarrow$
`</-`	$\not\longleftarrow$	`-/>`	$\not\longrightarrow$	`==`	$=$
`<=>`	$\rightleftharpoons$	`>=<`	$\leftrightharpoons$	`<>`	$\rightleftarrows$
`<=>>`	$\rightleftharpoons$	`>=<<`	$\leftrightharpoons$	`<==>`	$\Longleftrightarrow$
`<<=>`	$\rightleftharpoons$	`>>=<`	$\leftrightharpoons$	`<o>`	$\longleftrightarrow$
`<=o>>`	$\rightleftharpoons$	`<<=o>`	$\leftrightharpoons$	`<=o>`	$\rightleftharpoons$

Пакет chemformula вводит конструкцию `!(`*текст*`)(`*формула*`)`, печатающую *текст* под формулой:

`\ch{!(oxonium)( H3O+ )}` $\rightsquigarrow$ $\underset{\text{oxonium}}{H_3O^+}$, `\ch{!(water)(H2O)}` $\rightsquigarrow$ $\underset{\text{water}}{\text{H2O}}$.

Чтобы *формула* обрабатывалась правильно, ее нужно отделить от скобок пробелами. Чтобы «развалить» конструкцию и напечатать восклицательный знак перед круглой скобкой, отделите его парой фигурных скобок: `\ch{!{}(oxonium)( H3O+ )}` $\rightsquigarrow$!(oxonium)(H_3O^+).

Пакет chemformula позволяет настроить практически каждое действие верстки. Настройки конкретной формулы можно указать в параметре команды `\ch`, а для общих настроек определена команда `\setchemformula{`*настройки*`}`. Вдобавок к настройкам можно вводить новые связи, стрелки, изменять шрифты, выделять части формул цветом и т. д. Все это подробно описано в документации пакета [37], к которой отсылаем заинтересованных читателей. Здесь же осталось обсудить особенности его подключения.

Команда `\ch`, определяемая пакетом babel для печати гиперболического косинуса, конфликтует с одноименной командой пакета chemformula. Настраивая русскую верстку, пакет babel выдает ошибку, сообщая, что команда `\ch` уже определена. Пакет chemformula-ru [38] устраняет ее. Он сохраняет команду `\ch`, верстающую химические формулы, и делает неопределенной команду пакета babel. Его параметр cosh2ch позволяет настроить для печати гиперболического косинуса в русских нотациях стандартную команду `\cosh` (`$\cosh x$` $\rightsquigarrow$ $\operatorname{ch} x$), а параметр ch2Ch вводит для этого команду `\Ch` (`$\Ch x$` $\rightsquigarrow$ $\operatorname{ch} x$).

[10] Символу «`o`» соответствует строчная латинская буква «o».

4.9.1. Размерности физических величин

Строение сложных размерностей физических величин схоже со строением химических формул, поэтому для их печати очень удобно применять команду `\ch`, например

`\ch{A/см^2}` $\rightsquigarrow$ $\mathrm{A/см^2}$ или `\ch{A*см^{-2}}` $\rightsquigarrow$ $\mathrm{A \cdot см^{-2}}$.

Стандартная отбивка знака умножения слишком велика для размерностей, поэтому ее стоит уменьшить, сделав отрицательным пробел, окружающий по умолчанию этот знак:

`\ch[ adduct-space = -0.1em ]{A*см^{-2}}` $\rightsquigarrow$ $\mathrm{A \cdot см^{-2}}$.

Глава 5

Библиография и списки

Как правило, рукопись содержит какой-то список. Это может быть библиография к статье, предметный указатель в книге, или просто перечисление чего-либо. LaTeX располагает обширным набором средств оформления таких списков.

5.1. Списки в тексте

Списки, используемые непосредственно в тексте, делятся на три типа: нумерованные, ненумерованные и перечни описаний каких-то параметров, терминов, понятий и т. д. Они создаются с помощью окружений enumerate, itemize и description, имеющих одинаковый синтаксис, проиллюстрированный примером нумерованного списка:

```
\begin{enumerate}
  \item Текст первого пункта...
  Второй абзац первого пункта.
  \item Текст второго пункта...
  \item И так далее...
\end{enumerate}
```

1. Текст первого пункта...
   Второй абзац первого пункта.
2. Текст второго пункта...
3. И так далее...

Списки делятся на элементы, или записи, и могут содержать любое их количество. В свою очередь, записи могут состоять их нескольких абзацев текста. В сверстанном документе элемент помечается меткой, которая автоматически ставится перед ним. В рукописи каждая за-

пись начинается командой `\item[настройка метки]`. Действие ее параметра и вид меток определяются типом списка.

Списки разных типов можно вкладывать друг в друга, при этом глубина вложенности списков одного типа не должна превышать четырех уровней, а общая вложенность не может быть больше шести. Вместе с изменением уровня вложенности изменяются отбивки и метки записей. Покажем это на примере списков itemize:

```
\begin{itemize}
  \item Первая запись...
  \item Его вторая запись.
    \begin{itemize}
      \item Первая запись...
      \item Его следующая запись.
    \end{itemize}
  \item Вернулись из...
\end{itemize}
```

- Первая запись списка.
- Его вторая запись.
  - Первая запись вложенного списка.
  - Его следующая запись.
- Вернулись из вложенного списка в основной.

Ненумерованные списки формирует окружение `itemize`, не имеющее параметров и аргументов. Его записи помечаются символами-метками, набор которых определяется классом документа. В стандартных классах основной список помечается символом « • », а вложенные списки символами «–», «∗» и « · ». Параметр команды `\item` позволяет заменить метку отдельной записи. Например, запись `\item[+]`... будет помечена знаком «+». Глобальная замена меток описана в разд. 8.10.1.

Нумерованные списки верстает окружение `enumerate`, элементам которого присваиваются номера. Их генерируют счетчики записей, ассоциированные с каждым уровнем вложенности списка. Формат вывода номера зависит от уровня вложенности.

1. Элементы основного списка нумеруется арабскими цифрами.
   (a) Списка второго уровня — строчными латинскими буквами.
      i. Третьего уровня — строчными римскими цифрами.
         A. Список четвертого уровня нумеруется заглавными латинскими буквами.
2. При возврате на прежний уровень нумерация продолжается.

Номер элемента можно заменить другой меткой, указав ее в параметре команды `\item`. «Пропущенный» номер, получит следующий элемент. Для подавления вывода номера используйте пустой параметр. Если поставить в элементе метку `\label`, на его номер можно сослаться

командой \ref, например последний пункт приведенного вложенного списка имеет номер 1(a)iA.

Для верстки списка терминов, параметров и т. д. предназначено окружение description, использованное в описаниях классов документов (см. разд. 1.5.1), пакетов и их настроек (см. разд. 3.1.1), стилей и баз BibTEX в разд. 5.2.3. Предполагается, что в таком списке записи состоят из заголовка и поясняющего текста. По умолчанию заголовок, помещаемый в параметр команды \item[заголовок], выделяется жирным шрифтом, но стиль его печати легко изменить командами \text... или декларациями, которые не повлияют на печать поясняющего текста:

```
\begin{description}
  \item[Раки]...
  \item[\it Жабы]...
\end{description}
```

Раки обитают в водоемах, любят чистую воду. Бывают маленькими, большими и ну о-о-о-очень большими.

Жабы обитают в болотах, водоемах и прочих сырых местах. Бывают зелеными и не очень.

В отличие от itemize и enumerate, вложенность списков description ограничена не четвертым, а шестым уровнем.

5.2. Список литературы

Формируя список литературы, или библиографию, автор сталкивается с двумя принципиальными трудностями.

Во-первых, полиграфические традиции издательств определяют большое разнообразие стилей оформления библиографии, поэтому список литературы, подготовленный для какого-то издания, для другого, скорее всего, придется верстать заново. Верстка же даже небольшой библиографии требует напряженного внимания и занимает значительное время.

Во-вторых, чаще всего библиография представляет собой нумерованный список. В рукописи источники должны нумероваться в порядке появления, при этом их номера должны соответствовать порядку следования источников в библиографии. В процессе работы над рукописью порядок цитирования меняется очень часто, и нет смысла каждый раз приводить в соответствие ему список литературы. Однако даже по окончании редактирования сортировку библиографии, насчитывающей несколько десятков источников, сделать непросто. Чтобы обойти эту проблему в обзорных статьях, цитирующих большое количество работ,

источники часто сортируются по фамилии первого автора, используемой в качестве ссылки в тексте рукописи.

Обе проблемы эффективно решаются с помощью специальной программы, формирующей список литературы, в качестве которой обычно выступает компилятор BibTEX. Для него уже накоплены большие ресурсы, кардинально упрощающие верстку библиографии, поэтому мы разберем метод ее формирования совместными усилиями компиляторов LATEX и BibTEX.

Покажем сначала, как выглядят ссылка на источник и список литературы в рукописи:

```
.... в книге \cite{Knuth-TeXbook-1993} описаны ...

\begin{thebibliography}{99}
  \bibitem{Knuth-TeXbook-1993}
    Кнут~Дональд Е. \emph{Все про \TeX} --- Протвино: ...
```

а теперь, как они выглядят в документе:

... в книге [1] описаны...

Литература

[1] Кнут Дональд Е. *Все про TEX* — Протвино: АО RDTEX, 1993 — С. 575. — ISBN: 5-900614-01-8

5.2.1. Библиография и цитирование

Список литературы формирует окружение thebibliography:

`\begin{thebibliography}{`*текст*`}`.

Длина *текста* в его аргументе задает ширину поля, отводимого для меток. Для нумерованной библиографии в аргументе указывают целое число с количеством цифр не меньшим, чем у максимального номера в списке, например 99, если число источников не превышает сотни.

Записи списка начинаются командами `\bibitem[`*метка*`]{`*ярлык*`}`, за которыми следует описание источника литературы. Если команда `\bibitem` имеет параметр, указанная в нем метка выводится вместо

номера источника как в самом списке литературы, так и при ссылке на него. В аргументе команды `\bibitem{•}` указывают ярлык записи, служащий для идентификации источника при цитировании. Каждый ярлык должен быть уникален.

Для цитирования используют команду[1]

`\cite[`*дополнительная информация*`]{`*список ярлыков*`}` .

В ее аргументе через запятую перечисляют ярлыки.[2] Список ярлыков можно разделить пробелами, поставив их *после* запятых. Ставить же пробелы перед ними не следует, так как они могут вызвать сбой идентификации ярлыков.

В ходе компиляции LaTeX находит записи с требуемыми ярлыками и заменяет команды `\cite` номерами или метками источников.

Команда `\cite` имеет параметр, позволяющий уточнить ссылку. Чаще всего в него помещают номера страниц, томов, разделов и т. д. Например, уточненное цитирование приведенного выше источника `\cite[c.125]{Knuth-TeXbook-1993}` в документе примет вид [1, с. 125] и отошлет читателя к нужной странице обширного руководства по TeX.

Номера источников, перечисленных в аргументе команды `\cite`, не упорядочиваются. Чтобы сделать это, загрузите пакет cite , который не только следит за их порядком, но и сокращает список последовательных номеров до диапазона. Пакет имеет гибкие настройки и вводит множество новых команд для управления всеми аспектами цитирования, подробно описанными в его документации [39].

Загрузка пакета hyperref преобразует ссылки на источники в гиперссылки документа. Вдобавок к этому при использовании современных стилей верстки библиографии, например из коллекций gost, elsarticle, revtex, адрес, DOI или название источника преобразуются во внешнюю гиперссылку, если она указана в его описании.

5.2.2. Верстка библиографии с помощью BibTeX

Верстку библиографии следует доверить компилятору BibTeX, в качестве которого обычно выступает программа bibtex . Для этого нужно создать библиографическую базу данных и обеспечить взаимодействие компиляторов LaTeX и BibTeX, блок-схема которого приведена на рис. 5.1. Стрелками показаны направления потоков информации.

[1] Отметим, что команда `\cite` является хрупкой (см. с. 33).

[2] В TeXstudio при наведении курсора на ярлык всплывает окошко с описанием цитируемого источника, позволяющее проверить ссылку.

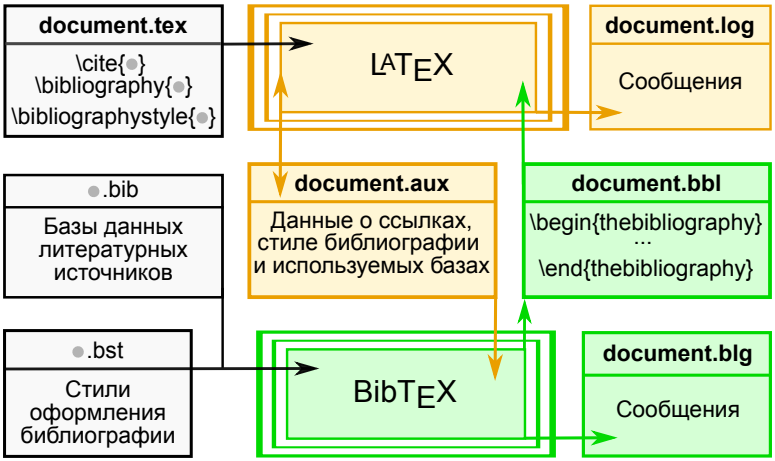


Рис. 5.1. Создание библиографии с использованием BibTeX

Рассмотрим пример, содержащий цитирование источников и команды, необходимые для взаимодействия компиляторов:

```
... \cite{Brandt-PRL-1993, Clem-PRB-1995} ...
    .....
\bibliographystyle{unsrt}
\bibliography{Superconductivity, CriticalState}
```

Обнаружив команду `\cite{●}`, компилятор LaTeX направляет содержащийся в ней список ярлыков в файл с расширением ●.aux. Команды `\bibliographystyle{●}` и `\bibliography{●}` заносят в него информацию о стиле верстки библиографии и библиографических базах, в которых находятся источники. Порядок следования ярлыков, устанавливаемый в ходе компиляции, автоматически соответствует очередности их появления в командах `\cite`.

Запускаемый следом компилятор BibTeX анализирует содержимое ●.aux-файла, загружает указанный стиль и библиографические базы, извлекает из них записи с ярлыками цитируемых источников, формирует окружение `thebibliography` и помещает его в файл с расширением ●.bbl.[3] При последующих компиляциях LaTeX загружает из него сформированный список литературы и разрешает ссылки.

[3] Напомним, что все автоматически генерируемые файлы наследуют имя компилируемого документа.

Если библиографию верстает BibTEX, для цитирования нужно использовать ярлыки, присвоенные источникам в библиографических базах. Если какай-то из источников не найден, BibTEX выдаст сообщение об ошибке, направив его в файл с расширением •.blg, а когда в источнике отсутствует информация, необходимая для правильного цитирования, например название журнала или книги, год опубликования и т. д., выдается предупреждение.

Для передачи списка библиографических баз компилятору BibTEX служит команда `\bibliography{`*список*`}`. В ее аргументе через запятую перечисляются имена файлов, имеющих стандартное расширение •.bib, которое можно опустить. Данная команда замещается сверстанной библиографией, поэтому ее положение определяет место списка литературы в документе. Перед отправкой в редакцию на ее место нужно скопировать окружение thebibliography из •.bbl-файла, а саму команду закомментировать или убрать. Все необходимое для генерации списка литературы в издательстве должна содержать сама рукопись.

Команда `\bibliographystyle{`*установка стиля*`}` задает стиль верстки библиографии. Ее аргументом является имя загружаемого стилевого файла, имеющего расширение •.bst, которое также можно опустить. Перед отправкой в редакцию эту команду тоже следует закомментировать или убрать.

Перечислим стандартные стили BibTEX, лежащие в основе других стилей верстки библиографии.

plain — данный стиль верстает нумерованный список литературы, в котором источники сортируются в алфавитном порядке по имени первого автора, затем по годам и по названиям публикаций.

alpha — этот стиль выполняет сортировку, аналогичную plain, но в библиографии и ссылках источники помечаются аббревиатурой, состоящей из первых букв фамилий трех первых авторов и последних двух цифр года выхода публикации, например JPG^{+}05. Знак + ставится, когда в публикации больше трех соавторов. Аббревиатуру единственного автора образуют первые три буквы фамилии.

abbrv — библиография верстается аналогично стилю plain, но имена авторов сокращаются до инициалов.

unsrt — стиль, верстающий список литературы, в котором источники не сортируются, а нумеруются в соответствии с порядком цитирования в рукописи.

Библиография документов, написанных по-русски, часто верстается по ГОСТ 7.0.5-2008. В соответствии с ним списки из четырех и менее соавторов выводятся без сокращения перед названием публикации, а более длинные списки урезаются до первых трех соавторов и печатаются за ним. Эти правила положены в основу стилей пакета gost [40]:

gost2008, ugost2008, gost2008l, ugost2008l — стили, верстающие библиографию, в которой источники следуют порядку цитирования в рукописи; стили с префиксом «u» предназначены для работы с базами, имеющими кодировку utf8; стили с окончанием «l» печатают перед названием публикации полный список авторов вне зависимости от их числа;

gost2008s, ugost2008s, gost2008ls, ugost2008ls — стили с окончанием «s» сортируют источники по фамилии первого автора или по названию публикации, если число соавторов больше четырех.

Для правильной обработки содержащиеся в базах русскоязычные источники должны иметь поле `language = {russian}` (см. далее). Более подробную информацию о перечисленных стилях содержит руководство пользователя пакета gost, подготовленное одним из его авторов И.А. Котельниковым [40].

При работе с библиографическими базами полезной оказывается команда `\nocite{`*список ярлыков*`}`, которая вносит источники в список литературы, не вставляя при этом ссылку в документ. Используя команду `\nocite{*}`, можно распечатать все содержимое баз, не цитируя их источники в рукописи.

5.2.3. Синтаксис библиографических баз

Как правило, страничка статьи на сайте журнала или в специализированных интернет-ресурсах типа Scopus или Web of Science позволяет загрузить ее выходные данные в формате BibTEX. Загрузка готовых записей дает наиболее эффективный метод формирования библиографических баз, имеющий свои преимущества и недостатки. К преимуществам нужно отнести автоматическое формирование описаний источников, гарантирующее отсутствие опечаток и ошибок. Отсюда же следуют и недостатки: алгоритмы автоматических процессов не способны охватить тонкости синтаксиса баз BibTEX, поэтому записи, полученные из интернета, часто требуют доработки. Разбирая синтаксис баз BibTEX, выделим моменты, на которые следует обратить внимание с этой точки зрения.

Библиографические базы представляют собой обычные текстовые файлы с расширением •.bib, содержащие описания источников в виде записей специального формата. Покажем для примера описание книги [3]:

```
@BOOK(Kotelnikov-LaTeX-2004,
  author = {Котельников, И. and Чеботаев, П.},
  title = "Настольная издательская система
           {\upshape\LaTeXe} по-русски",
  address = {Новосибирск},
  publisher = {Сибирский хронограф},
  year = 2004,
  pages = 491,
  isbn = {978-5-7931-0878-9},
  language = "russian")
```

Базы можно править в любом редакторе, однако использование специализированной программы ограждает от неосторожного удаления служебных символов, формирующих структуру баз, способного испортить не только отдельные записи, но и базу в целом. Для работы с базами BibTEX имеется свободно распространяемая программа JabRef, реализованная для всех операционных систем, доступная на сайте `https://www.jabref.org`. Она загружает записи из интернета и предоставляет удобный интерфейс для их просмотра и редактирования, описанный в разд. А.5.

Записи делятся на типы в соответствии с типом источника. Тип задается словом, начинающимся символом `@`, в котором заглавные и строчные буквы не различаются, например типы @ARTICLE и @article одинаковы. Перечислим стандартные типы записей.

@article — статья в периодическом издании.

@book, @inbook — книга и ее часть с указанным диапазоном страниц, номеров глав и т. д.

@incollection — часть книги, имеющая отдельное название.

@booklet — брошюра без указания издателя.

@conference, @proceedings — сборник трудов конференции.

@inproceedings — статья в сборнике трудов конференции.

@manual — техническая документация.

@masterthesis, @phdthesis — дипломная работа и диссертация.

@techreport — опубликованный отчет.

@unpublished — неопубликованный документ, имеющий автора и название, например статья в электронном архиве.

@misc — документ, не попадающий ни в одну из перечисленных категорий.

Запись содержит описание источника, которое заключается в фигурные или круглые скобки. Описание начинается с ярлыка, служащего для идентификации источника и его цитирования командами `\cite{•}`. Ярлык должен быть уникальным, легко запоминаемым и хорошо распознаваемым. Обычно автоматически генерируемые ярлыки содержат информацию, ценную для издателей или составителей баз, но бесполезную для авторов. Такие ярлыки неудобны в работе и их следует заменить в соответствие с вашими собственными предпочтениями. Как показывает опыт, оптимальным набором для ярлыка служит фамилия первого автора, аббревиатура издания и год опубликования работы, например `Ivanov-JETPL-2010`. Такой ярлык вы сразу вспомните, заглянув в публикацию.

За ярлыком следуют поля записи, разделяемые запятыми. Каждое поле состоит из названия и текста, разделенных знаком равенства. Текст заключается в двойные кавычки или фигурные скобки, которые можно опустить, если он сдержит только цифры. Так как текст записей служит основой формирования окружения `thebibliography`, он может содержать строчные формулы и команды форматирования LaTeX.

Перечислим названия и назначение стандартных полей.

address — адрес издательства или организации, опубликовавшей источник.

author, editor — список авторов или редакторов публикации.

booktitle — название книги или трудов конференции, содержащих публикацию.

chapter — номер главы, части или раздела книги.

doi — идентификатор публикации в сети Интернет.

eid — уникальный номер статьи в журнале, заменяющий номера страниц.

eprint — идентификатор публикации в сети или электронном архиве.

howpublished — поле, предназначенное для документов, способ издания которых не вписывается ни в какую другую категорию.

institution — организация, опубликовавшая отчет.

journal — название журнала.

key — слово, используемое при сортировке, если не указаны поля author или editor.

language — язык документа. Данное поле используется стилями gost2008. Для правильной обработки документов на русском языке оно должно иметь значение russian.

month, year — месяц и год опубликования или написания документа.

note — дополнительная информация, которая выводится вместе с выходными данными публикации.

number, volume — номер и том издания.

organization — организация, опубликовавшая труды конференции или техническую документацию.

pages — номер первой страницы, или диапазон страниц публикации.

school — учебное заведение, в котором написана диссертация или дипломная работа.

series — название серии книг или трудов.

title — название публикации.

type — уточнение некоторых параметров публикации.

url — адрес доступа к публикации в сети Интернет.

Поля делятся на обязательные, дополнительные и необязательные. Набор обязательных полей составляет информацию, необходимую для поиска публикации, зависящую от ее вида. Например, описание книги должно сдержать ее название, данные об издательстве и годе выпуска, тогда как для статьи нужно указать название, список авторов, название журнала и год его выпуска. Дополнительные поля хранят информацию, упрощающую поиск. В необязательных полях можно разместить комментарии и другую информацию, существенную с вашей точки зрения. Поля с нестандартными названиями, трактуются как необязательные. BibTEX предупреждает об отсутствии в записи обязательных полей, о дополнительных ничего не сообщает, а необязательные игнорирует.

В табл. 5.1 приведены стандартные типы записей и их обязательные и дополнительные поля. Часть из них имеет особый синтаксис.

Поле `author` содержит список авторов, разделяемый ключевым словом and, например `author = {Karl Marx and Frederick Engels}`. Обычное слово «and» в таком списке нужно заключить в фигурные скобки, например `author = "Procter {and} Gambel"`.

Таблица 5.1

Стандартные типы и поля библиографических записей[4]

Записи	author	title	journal	volume	number	year	month	pages	note	editor	publisher	address	isbn	edition	series	chapter	booktitle	organization	howpublished	school	institution	type	key	doi	url	eprint
@article	●	●	●	○	○	●	○	○	○														○	○	○	○
@book	∗	●		.	.	●	○		○	∗	●	○	○	○	○								○	○	○	○
@inbook	∗	●		.	.	●	○	⋆	○	∗	●	○	○	○	○	⋆						○	○	○	○	○
@incollection	●	●		.	.	●	○	○	○	○	●	○	○	○	○	○	●					○	○	○	○	○
@conference		●		.	.	●	○		○	○	○	○	○		○			○					○	○	○	○
@proceedings		●		.	.	●	○		○	○	○	○	○		○			○					○	○	○	○
@inproceedings	●	●		.	.	●	○	○	○	○	○	○	○		○		●	○					○	○	○	○
@booklet	○	●				○	○		○			○	○						○				○	○	○	○
@manual	○	●				○	○		○			○		○				○					○		○	○
@masterthesis	●	●				●	○		○			○								●		○	○		○	○
@phdthesis	●	●				●	○		○			○								●		○	○		○	○
@techreport	●	●			○	●	○		○			○									●	○	○		○	○
@unpublished	●	●				○	○		●														○		○	○
@misc	○	○			○	○	○		○										○				○		○	○

● обязательное поле, ∗ должно присутствовать одно из полей, ⋆ должны присутствовать
○ дополнительное поле, . может присутствовать одно из полей, одно или оба поля.

[4] Все записи могут содержать дополнительное поле language, записи @article и @misc — дополнительное поле issn. Записи @article также могут содержать дополнительное поле eid.

Как имя, так и фамилия могут состоять из несколько слов: Johann Sebastian Bach, Johannes Chrysostomus Wolfgangus Theophilus (Gottlieb) Mozart, Ludwig von Beethoven. Когда требуется сократить имя до инициалов (очевидно, фамилия не должна сокращаться), BibTeX использует следующий алгоритм ее определения. Если последовательность слов не содержит запятой, то фамилией является последнее слово. Фамилия, состоящая из нескольких слов, должна стоять перед именами и отделяться от них запятой: `von Beethoven, Ludwig`.

Если в описаниях авторов есть акцентированные буквы, а в записи, полученной из интернета, диакритические знаки отсутствуют, буквы следует заменить или поставить над ними акценты с помощью команд из табл. 2.9, например `Antonin Dvo\v{r}\'ak` ⇝ Antonin Dvořák.

В библиографии принято указывать сокращенные названия журналов, но в автоматически генерируемых записях они часто приводятся полностью, и тогда поле journal нужно отредактировать. Стандартное сокращение обычно указывается на сайте журнала, его также можно поискать в библиографиях цитируемых работ, а в программе JabRef его вставляет иконка справа от поля journal (см. разд. А.5).

Названия журналов можно заменить аббревиатурами, снижающими вероятность опечаток. Аббревиатуры определяют специальные записи @string{ *аббревиатура* = "*текст*"}. Введем к примеру аббревиатуру @string{ prb = "Phys. Rev.~B"} для сокращенного названия журнала Physical Review B. Аббревиатуры подставляются в качестве значения поля journal, но в отличие от обычного текста они не заключаются в скобки или кавычки. Вот как следует использовать нашу аббревиатуру: journal = prb. Обрабатывая запись, BibTeX заменит ее заданным текстом. Определения аббревиатур должны предшествовать их использованию, поэтому их следует размещать в начале баз.

Название публикации может содержать формулы химических соединений, простые математические формулы, собственные имена и другие слова, содержащие заглавные буквы. В автоматически генерируемых записях такие данные нужно внимательно отслеживать, так как часто они сделаны неправильно. В частности, это относится к полю title. Некоторые стили формируют из него предложение, которое начинается с заглавной буквы, а остальные буквы переводятся в строчные. Фигурные скобки помогают защитить текст от подобных преобразований. В них следует заключать химические формулы, собственные имена, географические названия и т. д., например

```
title = "Magnetic properties of {CeRu}\textsubscript{2}",
```

title = "Распределение {Бозе-Эйнштейна} в физике...".

Без такой защиты соединение $CeRu_2$ будет преобразовано в $ceru_2$, а фамилии тоже потеряют заглавные буквы.

Текст, помещаемый в фигурные скобки, обрабатывается «как целое». Он может находиться в любом поле, при этом в поле author скобки могут влиять на сортировку записей. Например, блок {von Beethoven} окажется в группе 'v', а фамилия von Beethoven попадет в группу 'B', так как BibTEX распознает приставку von как титул.

Поле type используется для уточнения или корректировки параметров ссылки. В записях @techreport в нем можно указать тип отчета, например type = "Отчет по гранту РНФ", в записях @phdthesis уточнить вид диссертации: type = "{Ph.~D} dissertation", а в записях @inbook и @incollection с его помощью корректируют уровень раздела. Например, если в источнике есть поля chapter и section, в ссылке на него будет указан номер главы, а после добавления поля `type = "section"` в ссылку попадет номер раздела. Отметим, что обработка поля type зависит от стиля библиографии и сказанное относится к стандартным стилям, поставляемым с BibTEX.

Поле crossref = {ярлык}, ссылающееся на ярлык другой записи той же базы, позволяет наладить перекрестные ссылки. Перекрестное цитирование помогает избежать дублирования одинаковых данных в материалах, относящихся к одному источнику. Например, поместив в запись @proceedings{SampleConf, ...} выходные данные трудов конференции, их можно опустить в записях @inproceedings, относящихся к докладам. В них достаточно указать авторов и названия докладов, диапазоны страниц и поле crossref = {SampleConf}, тогда недостающая информация будет добавлена из записи SampleConf. Записи, содержащие ссылки crossref, должны находиться *до* записи, на которую они ссылаются. Отметим, что BibTEX автоматически добавляет в библиографию источники перекрестных ссылок, на которые ссылаются более двух раз.

Загрузив и отредактировав запись, полезно проверить ее путем компиляции со стилем plain или unsrt. В названиях источников они преобразуют заглавные буквы в строчные, и потому наиболее чувствительны к ошибкам синтаксиса записей. Обычно они также позволяют «отловить» недопустимые unicode-символы (см. с. 60), часто встречающиеся в автоматически генерируемых записях.

Мы разобрали только основные особенности синтаксиса библиографических баз. Более полно они описаны в книгах [3, 4].

Глава 6

Набор рукописи

Правила оформления документов LaTeX не заданы жестко и однозначно, многие действия авторы выполняют, опираясь на собственный опыт, к сожалению, не всегда правильный. В этой главе кратко обсуждаются особенности верстки документов, написанных по-русски, и начинающим авторам даются рекомендации по набору рукописей. На некоторые вопросы мы уже обращали внимание ранее, но для полноты картины поднимем их здесь вновь.

6.1. LaTeX по-русски

Настройки, необходимые для редактирования текстов, написанных по-русски, выполняет пакет babel, загружаемый с параметром russian [41]. Он задает правила переноса слов, русифицирует стандартные названия, добавляет принятые у нас нотации тригонометрических и гиперболических функций и другие ресурсы верстки. Вместе с тем он не может полностью адаптировать LaTeX к традициям российской полиграфии, основные требования которой стоит обсудить особо.

Часть из них касается рисунков и таблиц. Автоматическое размещение в документе с помощью механизма плавающих объектов, применяемое LaTeX, плохо согласуется с необходимостью их расположения непосредственно перед или после первой ссылки на них, принятой в российских изданиях. Это требование можно обеспечить, используя при создании плавающих объектов параметр `h!`, а если это не помогает,

принудительно вывести их связкой команд `\afterpage\clearpage` (см. разд. 8.6).

В русских текстах номер и подпись к рисунку или таблице должны разделяться точкой, тогда как в стандартных классах для этой цели служит двоеточие. Здесь на помощь приходит пакет caption [42], позволяющий настроить все аспекты вывода подписей. В частности, чтобы уменьшить их шрифт и установить в качестве разделителя точку, загрузите его с параметрами:

```
\usepackage[font=small,labelsep=period]{caption}.
```

Особо стоит сказать о пояснениях к рисункам. В списках рисунков и в тексте подписи должны быть одинаковы, и чтобы не загромождать списки, их следует делать компактными. Если список не создается, как например в статьях, пояснения лучше дать непосредственно в подписях. В противном случае их можно внести в текст или привести в самих рисунках, как это сделано на рис. 3.6. ГОСТ Р 2.105-2019, регламентирующий верстку научно-технической документации, рекомендует размещать пояснительные данные над подписью в виде подрисуночного текста, тогда как в книгах пояснения дают в тексте.

Оформление подписей в списках и тексте может отличаться. При необходимости автоматическое разбиение подписей на строки можно корректировать командами `\\` или `\linebreak`, однако это следует делать только в аргументах команд `\caption`, в параметрах подписи указывают без дополнительного форматирования. Ранее говорилось, что в подписях могут находиться элементы, которые не должны попасть в списки, как например, ссылка на примечание в подписи табл. 5.1, или двоеточия, которыми оканчиваются подписи рис. 3.3–3.5. В таких случаях также помогают параметры команд `\caption`, в них подписи указываются без ненужных элементов. Отметим, что все сказанное относится также к параметрам команд разделов, служащим для внесения заголовков в оглавление.

Компактные таблицы являются наиболее удобным и эффективным методом организации справочного материала. В российской полиграфии принято отчеркивать колонки вертикальными линиями, однако это не является жестким требованием. Пункт 6.8.5 ГОСТ Р 2.105-2019 «Единая система конструкторской документации. Общие требования к текстовым документам» регламентирует следующее [43]: «Таблицы слева, справа и снизу, как правило, ограничивают линиями. Допускается не ограничивать таблицы линиями слева и справа. Разделять заголовки и подзаголовки боковика и граф диагональными линиями

не допускается. Горизонтальные и вертикальные линии, разграничивающие строки таблицы, допускается не проводить, если их отсутствие не затрудняет пользование таблицей.»

Чаще всего вертикальное отчеркивание загромождает таблицу и увеличивает ее ширину, поэтому большинство зарубежных издательств его не использует. В рукописях статей рекомендуется отчеркивать лишь верх, низ и заголовок таблиц. Так как данная книга посвящена оформлению научно-технической документации в LaTeX, мы отошли от правил российской полиграфии и использовали этот стиль оформления таблиц. Современные полиграфические технологии позволяют применять фоновое выделение колонок или строк, обсуждаемое далее в разд. 10.6, однако в журнальных публикациях это не принято.

Существуют особенности и в оформлении библиографии. Большинство стилей BibTeX, включая gost, заключают номера источников в скобки, тогда как по правилам российской полиграфии после номера ставится точка. Для соблюдения данного требования нужно отредактировать команду `\@biblabel`, печатающую номер, как описано на с. 159. Это лучше сделать в конце преамбулы документа, а если не сработает — непосредственно перед окружением thebibliography или генерирующей его командой `\bibliography`.

В ссылках на интернет-ресурсы следует указывать дату последнего обращения к ним, так как интернет-адрес может быть изменен, заблокирован или аннулирован. Для этого пакет gost вводит поле urldate, которое нужно добавить в записи баз BibTeX. Тогда в конце описания источников появится фраза «дата обращения» и сама дата.

Еще одним важным отличием является прямое начертание греческих букв в формулах, принятое в российской полиграфии, и курсивное в LaTeX. В данной книге использованы стандартные (для LaTeX) математические шрифты ComputerModern, в которых как латинские, так и греческие буквы печатаются курсивом. В свободном доступе имеются шрифты AMS Euler Американского математического общества, содержащие прямой математический курсив с начертанием и насыщенностью, близкой к шрифтам ComputerModern. Греческие буквы этих шрифтов подключает пакет upgreek (см. с. 148) коллекции was. При загрузке без параметра он подключит шрифты AMS Euler, а при загрузке с параметром Symbol будут использованы греческие буквы шрифта Adobe Symbol с начертанием и насыщенностью, близкими к шрифтам Times.

Буквы прямого начертания печатают команды с именами, начинающимися с префикса up, например `\upxi\,\xi` $\rightsquigarrow \upxi\,\xi$. Чтобы формулы,

содержащие прямые буквы, можно было бы копировать в другие документы, имеет смысл дать этим командам стандартные имена. Для этого удобно применить конструкцию TeX [1] `\let\B\A`, присваивающую команде `\B` значение команды `\A` (см. разд. 7.1). Загрузив пакет upgreek и сделав в преамбуле документа присваивания:

```
\let\alpha\upalpha  ...  \let\omega\upomega,
```

получим печать прямых греческих букв стандартными командами. Если не сделать переопределение, то при копировании может сложиться ситуация, в которой в документе окажутся формулы с буквами разных начертаний, что недопустимо. Заглавные греческие буквы переопределять не нужно, так как они имеют прямое начертание по умолчанию. Кардинальное решение проблемы греческих букв дает пакет unicodemath, обсуждаемый в разд. 12.3.6.

Последнее замечание касается обсуждавшегося ранее десятичного знака. Десятичную часть числа можно отделять либо точкой, либо запятой, но оба знака в одном документе с этой целью использовать нельзя! В LaTeX от десятичной запятой лучше отказаться, так как в числах она обивается как знак препинания, добавляя справа небольшой пробел. Рукописи статей должны содержать десятичную точку еще и потому, что в результате использования разных десятичных знаков в разных документах, копирование текста из одного документа в другой рано или поздно «смешает» их.

В российской полиграфии десятичным разделителем традиционно служит запятая. Действующий ГОСТ 1.5-2001 «Стандарты межгосударственные, правила и рекомендации по межгосударственной стандартизации. Общие требования к построению, изложению, оформлению, содержанию и обозначению» в п. 4.15.2 регламентирует «При записи десятичных дробей не допускается заменять точкой запятую, отделяющую целую часть числа от дробной». ГОСТ 8.417-2002 «Единицы величин» [44] пользуется только запятой, но сам по себе в качестве десятичного разделителя ее не устанавливает. ГОСТ 6.20.1-90 «Электронный обмен данными в управлении, торговле и на транспорте» и ГОСТ 2.004-88 «Общие требования к выполнению конструкторских и технологических документов на печатающих и графических устройствах вывода ЭВМ», а также международный стандарт ISO 80000-1:2009 допускают оба варианта.

Если правила оформления документа требуют, чтобы десятичная часть числа отделялась запятой, загрузите пакет icomma [45]. Он уберет ее отбивку в числах.

К верстке научно-технических отчетов и диссертаций ГОСТ 7.32-2017 и ГОСТ Р 7.0.11-2011 [46, 47] предъявляют дополнительные требования. Обсуждение настроек, обеспечивающих их выполнение, представлено в прил. Г, так как оно требует более глубоких знаний LaTeX, изложенных во второй части книги.

6.2. Полезные советы

Опыт преподавания LaTeX и чтения научной периодики, позволяет сформулировать несколько советов по набору рукописи.

Перед началом работы над документом, предназначенным для опубликования, внимательно ознакомьтесь с рекомендациями издателя. Автор должен понимать назначение пакетов, используемых в верстке, не стоит загружать их «по совету из Интернета». Ресурсы стандартного LaTeX и пакетов, рекомендуемых издателями, в большинстве случаев способны полностью обеспечить верстку. Значительное количество нестандартных пакетов уже не поддерживается, они могут конфликтовать друг с другом и обновляемыми стандартными средствами.

Заголовки разделов и подписи рисунков и таблиц должны быть информативными, но краткими. В их конце точку не ставят. Если они состоят их нескольких предложений, то в последнем точку опускают.

Хотя команды разделов стандартных классов не запрещают переносы слов, в заголовках они не допускаются. Чтобы избежать их, используйте команды `\\`, `\newline` или `\linebreak`. При этом не забудьте скопировать в параметры команд заголовки без принудительных разрывов, чтобы они не появились в оглавлении.

Не игнорируйте неразрывные пробелы. Ссылки и их номера, печатаемые командами `\ref` и `\pageref`, должны находиться в одной строке. То же касается значений и размерностей. Наряду с тильдой неразрывным является тонкий пробел «`\,`», который во многих случаях смотрится лучше. Сравните, например:

```
и т.~д., и т.\,п ⇝ и т. д., и т. п.,
см. рис.~\ref{...} на с.\,\pageref{...} ⇝ см. рис. 3.2 на с. 67,
номиналом в 1~Ω или 2\,Ω ⇝ номиналом в 1 Ω или 2 Ω.
```

Не пользуйтесь звездочкой * в качестве знака умножения и знаком деления ÷ для указания диапазонов. В тексте знак умножения печатают символы «·» (код `U+b7`, команда `\textperiodcentered`) и «×» (код `U+d7`, команда `\texttimes`), а диапазон задает символ

«–» (код U+2013, команда `\textendash`) или лигатура «--». Выражение «Ом · см» выглядит много лучше, чем «Ом*см», а для набора 3–7 не потребуется математический символ. Отбивку диапазона можно отрегулировать с помощью тонкого пробела: `3\,--\,7` ⇝ 3–7.

Формулы являются частью текста. В конце вынесенных формул ставят точки, даже если они не входят в какие-либо предложения. Формулы, состоящие из нескольких выражений, разделяют запятыми. Расшифровки переменных, входящих в формулы, должны находиться в тех же абзацах что и сами формулы, т. е. следующие за ними предложения не должны начинаться с отступов. Отступы не появятся, если между формулами и текстом отсутствуют пустые строки, однако для наглядности кода их лучше поставить, а текст начать командой `\noindent`, подавляющей отступ.

В строчных формулах желательно избегать дробей, генерируемых командами `\frac`, так как их числители и знаменатели оказываются мелкими: `\frac{a+b}{c+d}` ⇝ $\frac{a+b}{c+d}$. В данном случае дробь лучше формировать косой разделительной чертой: $(a+b)/(c+d)$. Сказанное касается также дробей в показателях экспонент, верстаемых в виде $\mathrm{e}^{-E/kT}$. Не нужно забывать, что функция-экспонента, равно как и число $\mathrm{e} = 2.718281828$, должны печататься прямым шрифтом `\mathrm`. Используйте команду `\operatorname` для печати функций, отсутствующих в LaTeX.

Не «подписывайте» переменные с помощью индексов: τ_{exp} или τ_{calc}. Такие переменные делают формулы «рыхлыми». Переменные следует помечать одним-двумя символами, меняя при необходимости их начертание: `\tau_e, \tau_\textsf{e}` ⇝ $\tau_e, \tau_{\mathsf{e}}$, и/или добавляя акцент: `\tau_e, \breve\tau_e, \hat\tau_e` ⇝ $\tau_e, \breve{\tau}_e, \hat{\tau}_e$.

Не используйте математические формулы для верстки химических. Наименования химических элементов являются текстом. Верстайте химические реакции и обозначения сложных соединений средствами пакета chemformula (см. разд. 4.9).

Часть II

Дополнительные возможности

Глава 7

Стандартные операции

В данной главе рассматривается обширный набор средств LaTeX, хотя и являющихся стандартными, однако практически не используемых в оформлении публикаций. Мы обсудим приемы работы с боксами, счетчиками и длинами, создание и редактирование команд, тонкости использования пружин и цветную верстку.

7.1. Создание и настройка команд

При верстке отчетов, диссертаций, дипломных работ и т. д., особенно если они написаны по-русски, зачастую возникает необходимость изменить некоторые команды, хранящие стандартные заголовки типа *Оглавление, Рис.* и т. п. В данном разделе обсуждаются средства, позволяющие это сделать, однако подавляющее большинство команд LaTeX вовсе не предназначено для внешнего вмешательства. Их лучше не трогать или делать это в крайнем случае, с большой осторожностью, изучив сначала основы программирования TeX и LaTeX [1–4] и описание ядра LaTeX [48, 49].

Средства создания и изменения команд предназначены в основном для документов «личного пользования». Используя декларации

```
\newcommand{команда}[n][параметр]{определение},
\renewcommand{команда}[n][параметр]{определение},
\providecommand{команда}[n][параметр]{определение},
\DeclareRobustCommand{команда}[n][параметр]{определение},
```

авторы могут переопределять уже существующие команды и вводить новые. Первая декларациях создает новую команду, вторая переопределяет уже существующую, а третья вводит новую команду, если она еще не определена, в противном же случае будет действовать существующая. Декларация `\DeclareRobustCommand` совмещает в себе действие деклараций `\newcommand` и `\renewcommand`: она либо определяет новую команду, либо заменяет существующую. Создаваемые ей команды не являются хрупкими (см. с. 33). По умолчанию аргументы создаваемых команд могут содержать несколько абзацев текста, а чтобы ограничить их одним абзацем, команды нужно вводить декларациями «со звездочкой»:

```
\newcommand*{команда}[n][параметр]{определение},
\renewcommand*{команда}[n][параметр]{определение},
\providecommand*{команда}[n][параметр]{определение}
\DeclareRobustCommand*{команда}[n][параметр]{определение}.
```

Использовать новую команду можно только после ее определения. Область действия деклараций ограничена группой, поэтому команды, созданные или измененные внутри группы, вне нее не действуют. Новые команды (изменения) следует вводить в преамбуле, тогда областью их действия будет весь документ.

Перечисленные декларации имеют одинаковый синтаксис. Разберем его на примере `\newcommand`. Первым аргументом является новая команда, а вторым — часть кода, который подставляется в рукопись при каждом ее выполнении. По умолчанию новая команда не имеет ни аргументов, ни параметров, но при необходимости ей можно присвоить до девяти аргументов, количество которых объявляется в первом параметре декларации `\newcommand`. В описании кода, выполняемого командой, аргументы обозначаются номерами `#1`, `#2...` Во втором параметре декларации `\newcommand` для аргумента `#1` можно задать значение по умолчанию, тогда он станет параметром новой команды. Создаваемая команда может иметь только один параметр.

Поясним сказанное примерами. Часто используемый пробел шириной `1em`, задаваемый командой `\quad`, оказавшись в начале или конце строки, игнорируется компилятором и потому работает не всегда:

```
\quad \rule{1ex}{1ex} \hfill \rule{1ex}{1ex} \quad \\
```


В данном примере квадратики, рисуемые командой `\rule`, заключены между пробелами и жесткой пружиной `\hfill`, расталкивающей по краям строки.

Введем новую команду `\newcommand\Quad{\hspace*{1em}}`, содержащую команду `\hspace*{1em}`, которую компилятор выполняет всегда, поэтому пробел `\Quad` появится, где бы он ни стоял:

```
\Quad \rule{1ex}{1ex} \hfill \rule{1ex}{1ex} \Quad \\
```

Видно, что теперь квадратики сдвинулись от краев строки на ширину пробела. Добавим возможность изменять ширину посредством параметра новой команды `\renewcommand\Quad[1][1em]{\hspace*{#1}}`. При ее использовании без параметра ширина пробела останется прежней, а мы утроим ее, воспользовавшись параметром:

```
\Quad[3em]\rule{1ex}{1ex}\hfill\rule{1ex}{1ex}\Quad[3em]\\
```

Ранее говорилось, что наборы команд, используемых в тексте и формулах, различаются. Если команда, предназначенная для верстки текста, содержит математические символы, их нужно вводить с помощью команды `\ensuremath{`*математические команды*`}`, автоматически создающую формулу, когда символ печатается в тексте. Воспользовавшись ей, создадим команду `\TO`, связывающую в наших примерах код с результатом его выполнения математическим символом `\leadsto`: `\newcommand\TO{\ensuremath{\leadsto}}`. Теперь можно написать `\TO $\sin x$` $\leadsto \sin x$. Если определить команду в виде строчной формулы: `\newcommand\TO{$\leadsto$}`, ее нельзя будет использовать в других формулах, так как `$(a\TO b)$` $\equiv$ `$(a$ \leadsto $b)$` и символ $\leadsto$ окажется *между* двумя формулами. Наш же вариант команды работает везде: `$(a\TO b)$` $(a \leadsto b)$.

Следующий пример связан с изменением стандартных заголовков, которые выводят команды, собранные в табл. 7.1. Среди них заголовок `\bibname` используют классы book.cls и report.cls, а `\refname` — article.cls.

Приведенные значения, используемые по умолчанию, изменяются с помощью декларации `\renewcommand`. Заменим, например, *Оглавление* на *Содержание*: `\renewcommand\contentsname{Содержание}`. Аналогично редактируются все остальные заголовки, при этом можно также настроить их вывод. Например, заголовок *Приложение*, измененный следующим образом,

```
\renewcommand\appendixname{%
    \vspace*{-2\baselineskip}\mbox{}\hfill Приложение},
```

сместится вверх и прижмется к правому краю страницы.

Таблица 7.1

Стандартные заголовки

Команды	Стандартные заголовки	
`\astractname`	Abstract	Аннотация
`\appendixname`	Appendix	Приложение
`\bibname`	Bibliography	Литература
`\refname`	References	Список литературы
`\contentsname`	Contents	Оглавление
`\figurename`	Figure	Рис.
`\listfigurename`	List of Figures	Список иллюстраций
`\tablename`	Table	Таблица
`\listtablename`	List of Tables	Список таблиц
`\indexname`	Index	Предметный указатель
`\chaptername`	Chapter	Глава
`\partname`	Part	Часть
`\proofname`	Proof	Доказательство

Пакет babel, заменяющий английские заголовки русскими, и некоторые другие пакеты выполняют свои настройки непосредственно перед началом обработки документа, инициируемым командой `\begin{document}`. Их нельзя изменить, разместив корректирующие команды в преамбуле, но вносить изменения в сам документ неудобно, поэтому ядро LaTeX вводит команду `\AtBeginDocument{`*код*`}`, выполняющую *код* после команды `\begin{document}`. Аналогичная команда `\AtEndDocument{`*код*`}` позволяет выполнить *код* в конце верстки вместе с командой `\end{document}`. Таким образом, стандартные заголовки нужно корректировать в аргументе команды `\AtBeginDocument`, которая должна находиться в преамбуле.

В качестве еще одного примера разберем устройство команд, верстающих дроби и биномиальные коэффициенты в формулах. Все они образованы из команды

`\genfrac{`*левый разделитель*`}{`*правый разделитель*`}{`*толщина линии*`}{`*стиль*`}{`*числитель*`}{`*знаменатель*`}`,

формирующей *числитель* и *знаменатель*, разделенные линией и охваченные скобками. Числитель и знаменатель верстаются в *стиле*, задаваемом числами:

0 — `\displaystyle`, 1 — `\textstyle`,
2 — `\scriptystyle`, 3 — `\scriptscriptystyle`.

Если этот аргумент пуст, размер числителя и знаменателя определяется из контекста. Если же пуст аргумент, задающий толщину линии, используется толщина, установленная по умолчанию, а чтобы убрать линию, указывается нулевая толщина. С ее помощью дроби и биномиальные коэффициенты вводятся как

```
\newcommand\frac[2]{\genfrac{}{}{}{}{#1}{#2}},
\newcommand\tfrac[2]{\genfrac{}{}{}{1}{#1}{#2}},
\newcommand\binom[2]{\genfrac{(}{)}{0pt}{}{#1}{#2}}
```

Определения дробей различаются только значением аргумента стиль. Так как в определении `\frac` он пуст, размер дроби изменяется в зависимости от контекста и типа формулы, а для дроби `\tfrac` установлен размер `\textstyle`. В команде `\binom` подавлена печать разделяющей линии. Используя `\genfrac`, расширим набор биномиальных конструкций, заключив их в скобки:

```
\providecommand\bbinom[2]{\genfrac{[}{]}{0pt}{}{#1}{#2}},
\providecommand\Bbinom[2]{\genfrac{\{}{\}}{0pt}{}{#1}{#2}},
```

`\bbinom{a}{b}` $\rightsquigarrow \begin{bmatrix} a \\ b \end{bmatrix}$, `\Bbinom{a}{b}` $\rightsquigarrow \begin{Bmatrix} a \\ b \end{Bmatrix}$.

Какой-то из загружаемых пакетов уже мог ввести команды `\bbinom`, `\Bbinom`, поэтому мы определили их декларацией `\providecommand`. А теперь задействуем все возможности `\genfrac`:

```
\newcommand\ESWF[3][]{\genfrac{|}{\rangle}{0pt}{#1}{#2}{#3}}
```

`\ESWF[0]{\psi_1}{\psi_2}` $\rightsquigarrow \left|\begin{matrix} \psi_1 \\ \psi_2 \end{matrix}\right\rangle$.

Созданная команда имеет параметр, который по умолчанию пуст. Он настраивает размер верстаемого выражения, заданный в примере нулем (`\displaystyle`). Пара аргументов служит для ввода верхнего и нижнего выражений. Получившаяся конструкция обозначает волновую функцию «запутанных» состояний ψ_1 и ψ_2. Альтернативная конструкция для печати такой волновой функции приведена на с. 247, а еще один пример создания команды, имеющей аргументы и параметр, обсуждается на с. 207.

Создавая новые команды, важно правильно организовать область их действия. Определим команду, печатающую имя команды:[1]

```
\newcommand\cmd[1]{\texttt{\textbackslash#1}}, \cmd{test} ⇝ \test
```

[1] Обычно для этого используют стандартную команду `\verb` (см. с. 62), но она не работает в аргументах других команд, что не очень удобно.

Если мы хотим, чтобы имя всегда печаталось прямым шрифтом средней контрастности, такое определение не годится, так как оно задает только семейство шрифта, не меняя его начертание и насыщенность. Используя декларацию `\tt`, устанавливающую все нужные нам атрибуты шрифта, сократим код команды, ограничив действие декларации группирующими скобками, `\newcommand\cmd[1]{{\tt\textbackslash#1}}`. Если определение команды содержит декларации, автор должен сам отрегулировать область их действия. В некоторых случаях организация групп с помощью фигурных скобок может влиять на расстановку пробелов, поэтому в коде LaTeX группы формируют команды TeX `\begingroup` и `\endgroup`. Используем их в окончательном варианте определения нашей команды:

```
\newcommand\cmd[1]{\begingroup\tt\textbackslash#1\endgroup}.
```

Командами `\begingroup` и `\endgroup` можно пользоваться также для группирования текста, особенно при большом его объеме, когда велика вероятность потерять одну из группирующих скобок.

Ранее говорилось, что компилятор «съедает» пробел, следующий за именем команды, поэтому текст или символы, печатаемые командами, не имеющими аргументов и параметров, не отделяются пробелом от последующего текста:

```
\renewcommand\cmd{test}
```

`\cmd , ` ⇝ test, `\cmd тест` ⇝ testтест.

Это вполне оправданно, когда за такой командой следуют знаки препинания, но чтобы отделить пробелом обычный текст, после нее придется ставить два пробела или команду «`\ `», что неудобно. Если же добавить пробел в определение команды, она будет печатать его всегда, в том числе и перед знаками препинания, что неправильно. Проблему корректной расстановки пробелов решает команда `\xspace`, определяемая одноименным пакетом `xspace` [50]. Она печатает пробелы, если только за ней не следуют знаки препинания, одна из фигурных скобок, закрывающая круглая скобка, прямой или обратный слэш:

```
\renewcommand\cmd{test\xspace}
```

`\cmd , ` ⇝ test, `\cmd тест` ⇝ test тест.

Мы разобрали пользовательский уровень настройки команд LaTeX. На системном уровне для этих целей привлекают средства TeX [1]. В ряде случаев они оказываются полезны, поэтому рассмотрим вкратце и их. Для определения команд TeX использует команды[2]

[2] Команды `\def` и `\gdef` имеют более гибкий синтаксис, для анализа которого нужны глубокие знания TeX. Здесь описан его простейший вариант.

`\def` *команда* • `{`*определение*`}`, и `\gdef` *команда* • `{`*определение*`}`, где *команда* — имя новой или переопределяемой команды; • — список аргументов, используемых в *определении*. Имя команды набирается без скобок. Список может отсутствовать или содержать до девяти аргументов в форме `#`*номер* (например, `#1#2#3`), либо перечисляемых без скобок и запятых, либо разделяемых символами, которые зададут синтаксис определения аргументов создаваемой команды (см. далее).

Существенным отличием команд `\def` и `\gdef` является отсутствие механизма контроля LaTeX. Уже имеющиеся команды переопределяются без предупреждений и сообщений об ошибке. Действие команд, определяемых `\def`, ограничено группой, в которой они создаются, а команды, вводимые `\gdef`, являются глобальными, область их действия не ограничена ничем.

Имена команд более низкого уровня, чем макрокоманды, с которыми имеют дело авторы, часто содержат символ `@`, служащий защитой от их случайного изменения. Чтобы редактировать такие команды, нужно сначала с помощью декларации `\makeatletter` дать символу `@` статус буквы, а внеся изменения, вновь вернуть статус символа декларацией `\makeatother`.

Поясним сказанное практическим примером. В стандартных списках литературы номера источников заключаются в квадратные скобки, тогда как в традициях российской полиграфии от описаний источников их отделяет точка. Печатает номер команда `\@biblabel{`*метка*`}`. Напрямую ее редактирование недоступно, но применив конструкцию `\makeatletter \def\@biblabel#1{#1.} \makeatother`, получим команду `\@biblabel{•}`, печатающую номер (метку), завершая его точкой.

Конструкция `\let\B=\A` позволяет *присвоить* команде `\B` *действие* команды `\A`. Команда `\B` автоматически наследует аргументы и параметры команды `\A`. На момент присваивания она может быть не определена, но `\A` должна иметь значение. Если после присвоения команда `\A` изменяется, `\B` останется неизменной. Конструкция `\let\B=\A` унаследована из TeX. Знак равенства в ней часто опускают. В качестве примера введем синоним громоздкой команды, печатающей обратный слэш: `\let\tbs\textbackslash`, и воспользуемся этой более компактной командой: `\tbs` ⇝ \. Более практичное применение конструкции `\let` рассмотрено на с. 149 прил. Г.1.

Обсудим теперь, как сделать «команду со звездочкой», имеющую аргумент и параметр. В именах команд, определяемых декларациями `\newcommand`, допускаются только латинские буквы, а звездочка вос-

принимается как признак окончания имени. В ядре LaTeX определена логическая конструкция `\@ifstar{верно}{неверно}`, исполняющая аргумент верно, если символом является звездочка, и аргумент неверно — в противном случае. Таким образом, конструкция

```
\makeatletter \newcommand\cmd{\@ifstar\@cmd\cmd@} \makeatother
```

создаст команду `\cmd`, выполняющую команду `\cmd@`, и команду `\cmd*`, выполняющую `\@cmd`. Воспользовавшись еще одной логической конструкцией, `\@ifnextchar{символ}{верно}{неверно}`, добавим нашей команде проверку, вызывается ли она с параметром или без него:

```
\makeatletter   \newcommand\cmd{%
                \@ifstar{\@ifnextchar[ \@@cmd \@cmd}%
                        {\@ifnextchar[ \cmd@@ \cmd@}}   \makeatother
```

Теперь за проверкой звездочки следует поиск открывающей квадратной скобки, являющейся признаком параметра. В итоге обеих проверок образуется следующая связь команд:

`\cmd` ≡ `\cmd@`, `\cmd*` ≡ `\@cmd`, `\cmd[` ≡ `\cmd@@`, `\cmd*[` ≡ `\@@cmd`.

Обратим внимание на знаки комментария в определении команды `\cmd`. Они подавляют «паразитные» пробелы, которые могут появиться в результате ее выполнения или повлиять на разбор ее синтаксиса компилятором.

Осталось определить исполняемые команды. Пусть `\cmd`, как и прежде, печатает команду: `\cmd{sample}` ⇝ `\sample`, а если она хранит символ, то команда со звездочкой дополнительно печатает и его: `\cmd*{maltese}` ⇝ `\maltese` (✠). В параметр поместим комментарий и напечатаем его в круглых скобках: `\cmd[шрифт]{tt}` ⇝ `\tt` (шрифт), `\cmd*[amssymb]{oint}` ⇝ `\oint` (∮ amssymb). Таким образом,

```
\makeatletter \newcommand\cmd{%
   \@ifstar{\@ifnextchar[ \@@cmd \@cmd}%
           {\@ifnextchar[ \cmd@@ \cmd@}}
  \def\cmd@#1{\begingroup\tt\textbackslash#1\endgroup}
  \def\@cmd#1{\cmd@{#1} (\ensuremath{\csname #1\endcsname})}
  \def\cmd@@[#1]#2{\cmd@{#2} (#1)}
  \def\@@cmd[#1]#2{\cmd@{#2} (\ensuremath{\csname #2\endcsname} #1)}
\makeatother
```

Разберем код команд, создаваемых декларациями `\def`. Команда `\cmd@` печатает имя. Остальные команды используют ее с этой целью и дополнительно печатают в круглых скобках комментарий и/или символ. Устройство команды, полностью аналогичной `\cmd@`, но определенной декларацией `\newcommand`, подробно обсуждалось ранее. Символ печа-

тает команда \ensuremath. Находящаяся в ее аргументе последовательность \csname имя \endcsname превращает имя в исполняемую команду. Отметим, что \ensuremath{•} прекрасно справляется с печатью как математических, так и текстовых символов. Команда \@@cmd выводит символ и комментарий, а \cmd@@ — только комментарий.

Определения двух последних команд содержат по два аргумента: один — с именем, а другой — с комментарием. Ранее говорилось, что аргументы команд заключаются в фигурные скобки, если же это не сделано, компилятор берет в их качестве символы, следующие за именем команды, по одному на аргумент. Это справедливо в большинстве случаев, однако для команд, вводимых декларациями \def, TEX допускает более свободный синтаксис. Их аргументы могут разделяться любыми символами. Например первый аргумент команды, определенной как \def\command[#1](#2)|3|#4#5{...}, заключен в квадратные скобки, второй — в круглые, а третий — между знаками модуля. Точно также они должны вводиться и при ее использовании. Четвертый и пятый аргументы подчиняются общим правилам, они могут заключаться в фигурные скобки, или наследоваться из последующих символов.

На уровне пользователя, мы создали команду \cmd с параметром и аргументом, хотя формально сама она аргументов не имеет и лишь инициирует исполнение других команд. Все, что находится за ней, служит *их* аргументами. Первый аргумент команд \@@cmd и \cmd@@, содержащий комментарий, мы заключили в квадратные скобки, чтобы он выглядел, как параметр команды \cmd, а их второй аргумент — имя, заключенное в фигурные скобки, выглядит как ее аргумент.

7.1.1. Условные конструкции

Набор низкоуровневых команд, две из которых рассмотрены ранее, позволяет управлять действиями компилятора TEX в зависимости от выполнения различных условий. Стандартный пакет ifthen [51] расширяет эти возможности. Для этого он определяет две условных конструкции:

\ifthenelse{условие}{блок да}{блок нет},
\whiledo{условие}{тело цикла}.

Обе команды проверяют условие, значениями которого являются логические единицы true (верно) и false (неверно). Если оно верно, команда \ifthenelse выполняет блок «да», а в противном случае — блок

«нет». Команда `\whiledo` при верном условии выполняет *тело цикла* и вновь возвращается к его проверке до тех пор, пока условие станет неверным.

Элементарными проверками являются операции:

`=, <, >` — сравнение целых чисел, например `\value{•}<5`;

`\isodd{`*число*`}` — определение четности, дающее значение true для нечетных чисел;

`\lengthtest{`*длина1*`•`*длина2*`}` — сравнение жестких длин (здесь символ «•» означает операторы `=, <, >`);

`\equal{`*строка1*`}{`*строка2*`}` — сравнение строк;

`\boolean{`*переменная*`}` — определение значения логической переменной.

С помощью логических операций `\and` (и), `\or` (или), `\not` (отрицание) и группирующих команд-скобок `\(` и `\)` из элементарных проверок можно построить комплексное условие, например проверку того, что текущая страница имеет четный номер, а ее лист — альбомную ориентацию:

```
\whiledo{ \( \lengthtest{\paperheight < \paperwidth} \and
              \not\isodd{\value{page}} \) \or ... }{ тело цикла }.
```

Стоящие в конце точки и операция `\or` обозначают возможные дополнительные проверки.

В проверке эквивалентности строк используется их внутреннее представление, которое у двух строк, выглядящих при печати одинаково, может различаться. Например, отрицательный результат даст проверка `\equal{\IeC{\cyra}}{a}`, так как буква «а» и ее командное представление `\IeC{\cyra}` не эквивалентны.

Декларации `\newboolean{`*имя*`}` и `\provideboolean{`*имя*`}` создают логические переменные. Вторая делает это, только если такой переменной еще нет. Имя переменной не является командой, поэтому оно не должно содержать символ «\». Значение переменной устанавливает декларация

`\setboolean{`*имя*`}{`*значение*`}`.

Его можно задать явно, указав true или false, или с помощью условия. В этом случае переменной присваивается результат анализа условия. Например, команда

```
\setboolean{pagerange}{ \value{page}>49 \and \value{page}<101 }
```

задаст переменной pagerange значение true в диапазоне страниц от пятидесяти до ста и значение false — для остальных страниц.

В ядре LaTeX определена еще одна условная конструкция, которая проверяет наличие файла:

```
\IfFileExists{имя файла}{блок да}{блок нет}
```

Если файл найден, выполняется блок «да», а если отсутствует — блок «нет». Файл ищется не только в рабочей папке, но и во всех папках, доступных компилятору с помощью средств поиска, описанных в прил. А.

7.2. Новые окружения

Перед обсуждением создания и переопределения окружений оговоримся, что редактировать стандартные окружения крайне опасно, так как последствия могут оказаться непредсказуемыми. При необходимости лучше создать новые и пользоваться и ими.

Новые окружения вводят декларации

```
\newenvironment{имя}[число][значение]{перед}{после},
\newenvironment*{имя}[число][значение]{перед}{после},
```

а существующие переопределяют декларации

```
\renewenvironment{имя}[число][значение]{перед}{после},
\renewenvironment*{имя}[число][значение]{перед}{после}.
```

Область действия создаваемых окружений может содержать несколько абзацев текста, но если их имя содержит звездочку, оно ограничено одним абзацем. Синтаксис всех деклараций одинаков. Их первым аргументом является имя окружения. В первом параметре указывается число аргументов окружения, которое не должно быть больше девяти. Если окружение имеет параметр, его значение по умолчанию заносится во второй параметр, а в определении окружения он выступает как аргумент `#1`.

В аргументы перед и после помещается код, который при верстке замещает команды `\begin{•}` и `\end{•}` и выполняется перед и после обработки области действия окружения.[3] В коде можно вводит новые

[3] Вместо кода эти аргументы могут содержать обычный текст, выводимый перед или после текста окружения.

команды, чтобы использовать их в окружении. Эти команды могут иметь собственные аргументы, однако аргументы окружения им недоступны. При выходе из окружения они становятся неопределенными. Действие команд, находящихся в коде перед, распространяется на всю остальную часть окружения, включая код после.

Создадим для примера список Itemize с настраиваемыми метками:

```
\newenvironment{Itemize}[1][]
    {#1 \begin{itemize}} {\end{itemize}},
```

Он верстает список itemize, выполняя перед этим команду, заданную параметром. По умолчанию параметр пуст, и в этом случае списки Itemize и itemize идентичны. Введем декларацию, настраивающую печать меток:

```
\newcommand\itemlabels[4]{
    \renewcommand\labelitemi{\labelitemfont #1}
    \renewcommand\labelitemii{\labelitemfont #2}
    \renewcommand\labelitemiii{\labelitemfont #3}
    \renewcommand\labelitemiv{\labelitemfont #4}  }
```

Ее аргументами являются символы меток четырех уровней записей (см. разд. 8.10.1). Сверстаем список Itemize, настроив метки:

```
\begin{Itemize}[\itemlabels •*∘·]
   \item Первый уровень...
   \begin{itemize}
      \item Второй уровень...
      .....
\end{Itemize}
```

- Первый уровень •
  - Второй уровень –
    - Третий уровень ∗
      - Четвертый уровень ·

Слева записи помечены настроенными метками, а справа для сравнения приведены стандартные. Мы заменили метки второго и третьего уровня, при этом декларация `\itemlabels` применена в окружении первого уровня, так как вносимые ей изменения действуют и во вложенных списках. Поместив ее в тесте рукописи, можно настроить метки всех следующих за ней списков itemize. Находясь в преамбуле, она настроит метки всего документа.

Были рассмотрены далеко не все тонкости определения команд и окружений, но этого краткого введения достаточно для настройки большинства конструкций LaTeX, с которыми сталкиваются авторы. За более подробной информацией заинтересованный читатель может обратиться к книгам [1–4].

7.3. Операции с боксами

Боксы, представляющие собой прямоугольники определенной ширины, высоты и глубины, — основные элементы верстки. Они могут содержать отдельные символы, несколько абзацев текста, формулы, рисунки, таблицы, а также другие боксы. Пожалуй, единственным ограничением является то, что боксы нельзя разбить на страницы.

В предыдущей части уже обсуждалось создание простейших боксов командами `\mbox` и верстка министраниц со сложной структурой окружениями `minipage`. Здесь мы рассмотрим операции, предназначенные для манипулирования боксами.

Команда `\makebox[`*ширина*`][`*выравнивание*`]{`*текст*`}` создает однострочный бокс, ширину которого можно зафиксировать. Без параметров она подобно команде `\mbox` формирует бокс с шириной равной длине *текста*, а когда задана *ширина*, компилятор использует ее при встраивании бокса в строку. Если текст длиннее объявленной ширины, он наложится на окружающий текст, если его длина меньше ширины бокса, он будет выровнен по методу, заданному вторым параметром:

`c` — центрирование, `s` — выравнивание с обеих сторон,
`l` — выравнивание слева, `r` — выравнивание справа.

Если же второй параметр опущен, текст центрируется.

Приведем примеры выравнивания текста в боксах и его взаимодействия с окружающим текстом, границы которого обозначим вертикальными линиями, а сам текст — вычеркивающими линиями. Слева и справа в примерах показаны боксы с коротким и длинным текстом, а в центре — верстающий их код:

```
|  инь и ян  | |\makebox[6em][c]{•}| /////|единство противоположностей|/////|
|инь и ян    | |\makebox[6em][l]{•}| |единство противоположностей|/////////|
|    инь и ян| |\makebox[6em][r]{•}| /////////|единство противоположностей|
|инь  и  ян| |\makebox[6em][s]{•}| |единство противоположностей|
```

Как видно из примеров, части строк, выступающие за границы боксов, налагаются на окружающий текст, представленный штриховкой.

С помощью длин `\width`, `\height`, `\depth` и `\totalheight` в параметре команды `\makebox` доступны значения ширины, высоты и глубины бокса и его полной высоты, равной сумме высоты и глубины. Сравните `\fbox{i}` ⇝ i и `\fbox{\makebox[\height]{i}}` ⇝ i. Для наглядности буква «i» помещена в рамку, которую рисуют команды `\fbox{`*текст*`}` и `\framebox[`*ширина*`][`*выравнивание*`]{`*текст*`}`,

действующие аналогично боксам `\mbox` и `\makebox`. Длины `\fboxrule` и `\fboxsep` хранят толщину рамки и ширину поля, отделяющего ее от *текста*.

Команда `\raisebox{`*сдвиг*`}[`*высота*`][`*глубина*`]{`*текст*`}` позволяет сдвигать боксы вверх и вниз. Ее первый аргумент задает величину сдвига. Положительная длина соответствует сдвигу вверх, а отрицательная — вниз. Так как данная операция может раздвинуть строки, параметры позволяют скорректировать вертикальные размеры бокса. При встраивании в строку объявленные *высота* и *глубина* используются вместо его истинных размеров. Например, сдвигая букву «у» на ее высоту, команда `\raisebox{\height}[\height]{y}` ↝ $^{\text{у}}$ полностью маскирует сдвиг. Здесь мы воспользовались оригинальной высотой бокса, размеры которого доступны команде `\raisebox`.

Пакет graphicx вводит ряд обсуждаемых далее команд, позволяющих масштабировать и вращать боксы. Среди них команда

`\scalebox{`*масштаб*`}[`*вертикальный масштаб*`]{`*текст*`}`

масштабирует бокс относительно его собственных размеров. Ее первым аргументом является действительное число, задающее коэффициент измерения размера, например `\scalebox{1.25}{текст}` ↝ текст. Число может быть отрицательным, тогда вместе с масштабированием производится еще и отражение: `текст\scalebox{-1}{текст}` ↝ $_{\text{ʇǝʞɔʇ}}$. Параметр позволяет установить разные коэффициенты горизонтального и вертикального масштабирования: `\scalebox{1}[1.5]{текст}` ↝ т℮КСТ. Если он отсутствует, бокс масштабируется пропорционально.

Производной от `\scalebox` является использованная в примере рисунка в конце разд. 3.3 команда зеркального отражения:

`\reflectbox{`*текст*`}` ≡ `\scalebox{-1}[1]{`*текст*`}` .

Подогнать бокс под заданные размеры позволяют команды:

`\resizebox{`*ширина*`}{`*высота*`}{`*текст*`}`,
`\resizebox*{`*ширина*`}{`*полная высота*`}{`*текст*`}`.

Они отличаются тем, что в `\resizebox` второй аргумент определяет высоту бокса относительно базисной линии строки, а в `\resizebox*` — его полную высоту. Если при масштабировании необходимо сохранить пропорциональность, один из аргументов заменяется восклицательным знаком, например `\resizebox{1em}!{n}` ↝ $\mathbf{n}$. В аргументах данных команд доступны оригинальные размеры бокса. Воспользуемся этим и изменим предыдущий пример: `\resizebox{1em}{\width}{n}` ↝ $\mathbf{n}$.

Команда `\rotatebox[`*список параметров*`]{`*угол*`}{`*текст*`}` обеспечивает вращение бокса. Положительный и отрицательные углы задают, соответственно, вращение по часовой стрелке и против нее. Параметры, перечисляемые списком через запятую, позволяют настроить положение оси вращения и задать единицы измерения угла.

origin = *координаты* — положение оси вращения, заданное координатами, привязанными к границам бокса или базисной линии строки. Горизонтальная координата может принимать значения **l**, r и c, что соответствует левой, правой границе и середине ширины бокса. Вертикальная координата t, b и c привязывается к верхней, нижней границе и середине высоты бокса, или положению базисной линии строки **B**. По умолчанию ось вращения находится на левой границе бокса в точке его привязки к базовой линии: origin=lB, а значение origin=c соответствует вращению относительно его центра.

x = *длина* — смещение оси вращения, от левой границы бокса.

y = *длина* — смещение оси вращения, от нижней границы бокса.

units = *число* — единицы угла поворота. Число задает полный оборот, поэтому units=6.28319 соответствует отсчету угла в радианах.

Для сравнения приведем несколько примеров, показав в них базовую линию:

```
\rotatebox{180}{пример}                  ↝  — dəwdıu — ,
\rotatebox[origin=c]{180}{пример}        ↝  — dəwdıu — ,
\rotatebox[x=.15\width]{-10}{пример}     ↝  — пример — .
```

В последнем примере ось вращения смещена относительно ширины бокса, оригинальные размеры которого определены в параметре команды `\rotatebox`.

Бокс можно сохранить, чтобы использовать впоследствии. Для этого, прежде всего, нужно ввести команду, которая будет его хранить: `\newsavebox{`*команда*`}` , а затем записать в нее бокс командами:

```
\sbox{команда}{текст},
\savebox{команда}[ширина][выравнивание]{текст} ,
```

или окружением lrbox :

```
\begin{lrbox}{команда}  текст  \end{lrbox}.
```

Окружение отличается от команд тем, что удаляет пробелы, находящиеся перед и после *текста*. Кроме этого в нем можно использовать

команду `\verb` и окружение `verbatim`. Сохраненный бокс выводит команда `\usebox{`*команда*`}`. Поясним сказанное примером:

```
\newsavebox\Tiger
\sbox\Tiger{\includegraphics{tiger}}
\newcommand\tiger[1][.1]{\scalebox{#1}{\usebox\Tiger}}
\newcommand\regit[1][.1]{\scalebox{-#1}[#1]{\usebox\Tiger}}
\mbox{}\hfil \regit\ \tiger
```

Создав команду `\Tiger`, мы сохранили в ней изображение тигра, а затем ввели для его печати команды `\tiger[•]` и `\regit[•]`, задав зеркальное отражение изображения командой `\regit`. Чтобы изменять размер картинки, у команд печати определен параметр, по умолчанию уменьшающий ее в десять раз. В конце мы вывели обе картинки, разделив их пробелом и отцентрировав с помощью пружины. Отметим также, что команда `\tiger[•]` (сохраненный бокс) многократно использовалась и в других примерах книги.

7.3.1. Клей \hss и боксы нулевой ширины

Особый набор боксов позволяет совмещать различные символы или налагать одну часть текста на другую. Чтобы понять принцип их действия, разберем сначала несколько примеров.

Помимо пружин `\hfil` и `\hfill`, имеющих бесконечную растяжимость, существует клей `\hss`, обладающий растяжимостью и сжимаемостью жесткости fil, который стремится подогнать длину текста под размер бокса, в котором находится. Он склеивает границы текста и бокса — левую с левой, а правую с правой. Если длина текста меньше ширины бокса, он аналогично пружине `\hfil` расталкивает текст, в противном же случае сжимает его так, что части текста накладываются друг на друга. В качестве разъяснения приведем наглядные примеры из книги [2]:

```
\rule[-.5ex]{1em}\arrayrulewidth%
\hbox to 4em {Кот\hss Пес}%          ⇝  __Кот  Пес¯¯,
\rule[2ex]{1em}\arrayrulewidth,
...\hbox to 3em {...}...             ⇝  __КотПес¯¯,
...\hbox to 0em {...}...             ⇝  ПесКот.
```

Команда `\hbox to длина {`*текст*`}`, наследованная из TEX, аналогично команде `\makebox` создает бокс заданного размера, но в отличие от нее не выравнивает *текст*. Декларация `\arrayrulewidth` хранит стандартную толщину линии (см. разд. 10.2). Знаки комментария подавляют появление лишних пробелов. Линии, прочерченные слева и справа, показывают границы боксов.

В первом примере ширина бокса превосходит длину слов Кот и Пес, поэтому клей `\hss` расталкивает их. Во втором примере ширина бокса уменьшена и текст оказывается длиннее, поэтому он сжимается так, чтобы не выходить за границы бокса. В третьем примере бокс имеет нулевую ширину, как следствие, конец нижней линии оказывается в том же положении, что и начало верхней, а текст располагается между ними. Так как границы бокса совпадают, приклеенное справа слово Пес оказывается перед словом Кот, приклеенным слева. Строго говоря, набор команд, приведенный в последнем примере, неполон. До и после отчеркивающих линий добавлены пробелы, чтобы Пес и Кот не «наехали» на стоящую впереди стрелку и завершающую точку.

Перейдем теперь к боксам-накладкам, действующим аналогично третьему примеру:

Команды		Примеры		
`\llap{•}`	= `\hbox to 0pt {\hss •}`	`*|\llap{\_}|*`	⇝	*||*,
`\rlap{•}`	= `\hbox to 0pt {• \hss}`	`*|\rlap{\_}|*`	⇝	*||*,
`\clap{•}`	= `\hbox to 0pt {\hss • \hss}`	`*|\clap{\_}|*`	⇝	*||*.

Здесь символом • обозначен текст, помещаемый в бокс.

Команда `\llap` создает бокс нулевой ширины, в котором перед текстом находится клей `\hss`, склеивающий правые границы текста и бокса, совмещая их правой границей предшествующего текста. Это показывает первый пример, в котором символ «_» находится слева от первой вертикальной линии. В команде `\rlap` клей стоит после текста, приклеивая его к левой границе бокса, поэтому последующий текст «подтягивается» к его левой границе, и символ «_» в примере совмещается с правой вертикальной линией. В команде `\clap` пружи-

ны \hss, находящиеся в начале и конце бокса, совмещают границы предшествующего и последующего текста, центрируя относительно них текст бокса, как продемонстрировано в последнем примере.

Боксы нулевой ширины используются во многих конструкциях LaTeX. С их помощью можно, например, сделать пометку на полях:

Этот трюк выполнил набор команд. \\ \llap{\fbox{!}\quad}.

Подобные боксы определены и для формул, в которых они помогают настраивать громоздкие индексы. Их применение обсудим в разд. 9.5.

7.4. Операции со счетчиками

Авторам дано право вводить новые счетчики, а также изменять подчиненность и формат вывода существующих. Новый счетчик определяет декларация \newcounter{*счетчик*}[*внешний счетчик*] . Если внешний счетчик не указан, создается независимый счетчик. Подсчитываемые им объекты будут иметь сквозную нумерацию, как у страниц. Если указан внешний счетчик, новый счетчик будет для него подчиненным, при этом сам внешний счетчик может быть в подчинении у другого счетчика. Например, счетчик subsection, являясь внешним для subsubsection, подчинен счетчику section. Подчиненность счетчиков передается по цепочке до самого верха, т. е. до независимого счетчика. Значения подчиненных счетчиков могут сбрасываться при операциях с внешними счетчиками. Например, счетчик subsubsection сбрасывается при создании раздела \subsection и выше, так как является подчиненным для subsection, section и т. д.

Декларация \counterwithin{*счетчик*}{*внешний счетчик*} изменяет подчиненность счетчика. Вот так можно установить нумерацию формул по разделам: \counterwithin{equation}{section}. Декларация \counterwithout{*счетчик*}{*внешний счетчик*} отменяет подчиненность счетчика. Например, в классе book рисунки нумеруются по главам, но, добавив в преамбулу рукописи декларацию \counterwithout{figure}{chapter}, можно установить их сквозную нумерацию.

Значения счетчиков обновляет команда \stepcounter{*счетчик*} . Она увеличивает значение указанного *счетчика* на единицу и сбрасывает в нуль подчиненные ему счетчики. С метками значения счетчиков связывает команда \refstepcounter{*счетчик*} . Она вызывает \stepcounter для обновления счетчика и затем изменяет системную

команду `\@currentlabel`, в которую помещает его новое значение. Если после `\refstepcounter` поставить команду `\label`, метка ассоциируется с ним.

Как говорилось в разд. 1.5.6, используя команду `\the`*счетчик*, можно распечатать текущее значение счетчика. Для независимых счетчиков выводится только одно число, равное их значению. Перед значениями подчиненных счетчиков обычно выводятся значения внешних. Например, данная страница имеет номер `\thepage` ⇝ 171, а номер данного раздела: `\thesection` ⇝ 7.4. Для печати определено несколько форматов, приведенных в табл. 7.2. Их задают команды, аргументом которых является имя счетчика, представленное символом «•».

Таблица 7.2

Форматы вывода счетчиков

Форматы	Цифры	Форматы	Числа
`\arabic{•}`	0 – 9	`\Alph{•}`	A–Z
`\Roman{•}`	C, I, L, M, V, X	`\alph{•}`	a – z
`\roman{•}`	c, i, l, m, v, x	`\Asbuk{•}`	А–Я
`\fnsymbol{•}`	∗, †, ‡, §, ¶, ‖, ∗∗, ††, ‡‡	`\asbuk{•}`	а – я

Наиболее часто значения печатаются арабскими цифрами, реже всего — символами нижней ячейки второго столбца, обозначающими цифры от 1 до 9. Ноль в данном формате отсутствует. Форматы с римскими цифрами также не имеют нуля, но могут представить любое число, большее или равное единице. Форматы с латинскими и русскими буквами выводят числа, соответственно, от единицы до двадцати шести и двадцати восьми.

Чтобы изменить формат вывода счетчика, нужно переопределить команду `\the`*счетчик*, например

```
\renewcommand{\theсчетчик}{\Roman{счетчик}}
\renewcommand{\theсчетчик}{\theвнешний.\Roman{счетчик}}.
```

Первый вариант настраивает печать большими римскими цифрами независимого счетчика, а второй — подчиненного. Точку, обычно разделяющую значения счетчиков, можно заменить любым другим символом. Например, по умолчанию значение подчиненного счетчика рисунков отделяется от счетчика глав точкой и выводится цифрами: `рис.~\ref{f:DocPage}` ⇝ рис. 1.2. После переопределения его печати, `\renewcommand\thefigure{\thechapter-\roman{figure}}`, ссылка примет вид `рис.~\ref{f:DocPage}` ⇝ рис. 1-ii.

В разд. 1.5.6 говорилось, что команды `\setcounter` и `\addtocounter` изменяют значения счетчиков. Стандартный пакет calc [52] коллекции tools позволяет проводить в их аргументах вычисления с операциями сложения, вычитания, умножения и деления. Вычисления проводятся слева направо. Для группирования выражений используются круглые скобки. Операндами могут выступать числа и значения счетчиков, получаемые с помощью команды `\value{`*счетчик*`}`.[4] Так как значения счетчиков целочисленны, основное правило вычислений сводится к тому, что результат каждой операции становится целочисленным. В операциях деления дробная часть отбрасывается без округления, поэтому выражение `\value{page}/6` ⇝ `5/6` ⇝ 0 равно нулю, а выражение `\value{page}/2` ⇝ `5/2` ⇝ 2 дает двойку. Здесь и далее значение `\value{page}` равно пяти. В цепочке вычислений дробная часть отбрасывается при каждой операции, поэтому выражение `2*\value{page}/6*5` ⇝ `(2×5/6)×5` ⇝ `(10/6)×5` ⇝ 5 равно пяти, а значение `2*(\value{page}/6)*5` ⇝ `2×(5/6)×5` ⇝ `2×0×5` ⇝ 0 зануляется. В вычислениях можно использовать вещественные числа, вводимые командой `\real{`*число*`}`. Их можно лишь умножать и делить, а прибавлять или вычитать нельзя. Результат произведения или деления целого и вещественного числа становится целым, поэтому выражение `\value{page}*\real{1.5}*\real{1.5}` ⇝ `5×1.5×1.5` ⇝ `7×1.5` ⇝ 10 равно десяти. Цепочки произведений и делений обязательно должны начинаться целым числом, например выражение `2*\real{1.5}` правильно, а `\real{1.5}*2` ошибочно. Практический пример вычисления счетчика мы обсудим вместе с операциями, производимыми с длинами.

В завершение работы со счетчиками рассмотрим настройку нумерации разделов. Как говорилось ранее, их уровни определяет числовой идентификатор, принимающий значения −1 для `\part`, 0 для `\chapter`, 1 для `\section` и т. д. до 5 для `\subparagraph`. Установка счетчика secnumdepth в одно из этих значений ограничивает максимальный уровень нумерации. Например, при установке его значения, равному двум, `\setcounter{secnumdepth}{2}`, разделы `\subsubsection` и ниже нумероваться не будут. Еще один счетчик tocdepth ограничивает максимальный уровень разделов, вносимых в оглавление. Если установить его в единицу, в оглавление попадут разделы уровня `\section` и выше. По умолчанию класс article устанавливает значения счетчиков secnumdepth и tocdepth равными трем, а классы book и report — двум.

[4] Отметим, что команда `\value` не печатает значения счетчиков, ее использование в этих целях вызовет ошибку.

7.5. Операции с длинами

Во многих случаях пружины — простой и удобный инструмент верстки, поэтому рассмотрим их устройство и возможности настройки.

Пружины являются пробелами с размерами, задаваемыми длинами. Как говорилось в разд. 2.1, длины могут иметь постоянную величину, изменять свой размер в определенных пределах, а также расширяться до любого размера. Они определяются основным размером и параметрами растяжимости и сжимаемости, которые указываются после ключевых слов plus и minus, например 12pt plus 2pt minus 3pt.

Новую длину вводит декларация `\newlength{команда}`. Созданной длине с помощью команды `\setlength` нужно присвоить значение. Основной размер задается обязательно, а параметры можно опустить, тогда размер длины фиксируется. Длина с параметрами становится упругой и может удлиняться на величину растяжимости или укорачиваться на величину сжимаемости. Основной размер и параметры могут быть положительны или отрицательны, но отрицательная растяжимость или сжимаемость вряд ли имеют практический смысл. Если основной размер равен нулю, например 0pt plus 4pt, то длина изменяется от нуля до величины растяжимости. Пробел, заданный упругой длиной, подстраивая свой размер, не расталкивает объекты. Такая длина не может служить материалом для «изготовления» пружин, так как не имеет жесткости.

Длина, заданная командой, используемая в какой-либо другой команде, утрачивает свойство растяжимости и сжимаемости. Например, пробел `\hspace{2\length}` с длиной `\length`, равной 12pt plus 4pt, будет иметь фиксированную ширину 24pt.

Величина длины определяется значением и размерностью. Для растяжимости пружин введена специальная размерность, которую мы назовем *жесткостью.* В LaTeX определены три единицы жесткости: fil, fill и filll. Чем больше букв «l» в размерности, тем больше жесткость. Команды `\hspace{0pt 1fil}` и `\hspace{0pt 1fill}` являются аналогами пружин `\hfil` и `\hfill`.

В виде команды LaTeX вводит единственную длину `\fill`, обладающую жесткостью, равной `0pt plus 1fill`. Определена также команда, `\stretch{число}`, позволяющая пользоваться длиной с растяжимостью, заданной долями жесткости fill. Ее аргумент — десятичное число, например пробел `\hspace{\stretch{2.5}}` является пружиной с жесткостью 2.5fill.

Пружины с одинаковым уровнем жесткости, т. е. одинаковой размерности, конкурируют между собой, а более жесткая сминает любую более мягкую. Покажем это на простом примере:

```
|\hfill|\hspace{0pt plus 2fill}|\hspace{0pt plus 100fil}| \\
```

| | ||

Между первыми вертикальными палочками вставлена пружина `\hfill` жесткостью 1fill, между второй и третьей — пружина жесткостью 2fill. Жесткость пружины, стоящей между третьей и четвертой палочками, составляет 100fil. Строка разорвана командой `\\`, неявно вставляющей в конце пружину `\hfil`. Из примера видно, что обе пружины с размерностью fil смяты, а расстояния между палочками пропорциональны жесткости пружин размерности fill.

Некоторые пакеты используют для выравнивания не мягкие, а жесткие пружины. Например, после загрузки пакета colortbl (см. разд. 10.6) ячейки таблиц выравниваются пружинами с жесткостью fill. В таких случаях для верстки может понадобиться более жесткая пружина, которую логично оформить в виде команды `\hfilll`:

```
\providecommand\hfilll{\hspace{0pt plus 1filll}}.
```

В нашем определении использована команда `\providecommand`, так как быть может какой-то пакет уже позаботился ввести недостающую пружину, или сделает это в будущем.

Следует отметить, что все сказанное выше в равной степени применимо и к вертикальным пробелам, и к пружинам.

Обсудим теперь вычисление значений длин, устанавливаемых командами `\setlength` и `\addtolength` (см. разд. 2.1). Это позволяет делать уже упоминавшийся пакет calc. Правила вычислений длин и счетчиков различаются. Значения длин задаются вещественными числами, которые в ходе вычислений не округляются.

Длины задаются явно или с помощью команд. В вычислениях могут участвовать длины разной размерности, например в выражении `\somelength -1mm` длина `\somelength` может иметь любую размерность. Длины можно складывать и вычитать, умножать на число, а также делить на число или другую длину. В качестве чисел могут выступать значения счетчиков.

В операциях умножения оператором `*` длина должна быть первым операндом, поэтому выражение `1ex*3` корректно, а `3*1ex` — нет.

Ранее многократно использовался способ умножения, назовем его «умножением слева», в котором число стоит перед длиной, а знак `*`

отсутствует. На примере упругих длин посмотрим, чем отличаются друг от друга два способа умножения:

```
\setlength\test{2pt plus 2pt minus 2pt}
\setlength\Length{2\test}      \the\Length ⇝ 4.0pt
\setlength\Length{\test*2}     \the\Length ⇝ 4.0pt plus 4.0pt minus 4.0pt
\setlength\Length{2\test*2}    \the\Length ⇝ 8.0pt
\setlength\Length{2\test +\test*2}
                               \the\Length ⇝ 8.0pt plus 4.0pt minus 4.0pt
```

Длина `\test` имеет растяжимость и сжимаемость. Длина `\Length` получается ее умножением на двойку разными способами. Видно, что умножение слева действует только на основной размер длины, при этом растяжимость и сжимаемость отбрасываются. Оператор `*` умножает каждую компоненту длины. В вычислениях можно комбинировать оба способа умножения.

В операциях умножения и деления на число длину, заданную явно, можно указывать без скобок:

```
2pt plus 1pt minus 2pt *2     ⇝     4.0pt plus 2.0pt minus 4.0pt.
2pt plus 1pt minus 2pt /2     ⇝     1.0pt plus 0.5pt minus 1.0pt.
```

Операторы `*` и `/` ставятся после последней ее компоненты. Операции с отдельными компонентами не поддерживаются, поэтому выражения `2pt *2 plus 1pt minus 2pt` и `2pt plus 1pt /2 minus 2pt` некорректны.

При умножении и делении на вещественное число растяжимость и сжимаемость упругой длины отбрасываются:

```
2pt plus 1pt minus 2pt *\real{1.5}     ⇝     3.0pt.
2pt plus 1pt minus 2pt /\real{1.5}     ⇝     1.33334pt.
```

С помощью команды `\ratio{длина}{длина}` две фиксированные длины можно поделить друг на друга. Результатом операции будет действительное число. Использование в числителе или знаменателе упругой длины даст ошибку.

Для примера вычислим, сколько четверостиший уместится на странице. Пусть в расчетах определены счетчик verse и длины `\numenator` и `\denumenator`. Сначала вычислим высоту одной строфы и вычтем ее из высоты текста на странице:

```
\setlength\numenator{\textheight - \baselinestretch\baselineskip*4}.
```

Напомним, что высоту строки задает произведение коэффициента `\baselinestretch` на длину `\baselineskip`.

Теперь вычислим высоту одной строфы с учетом пустой строки, разделяющей строфы, и вертикального пробела `\parskip`, вставляемого

между абзацами-строфами:

`\setlength\denumenator{\baselinestretch\baselineskip*5 + \parskip}`.

И наконец, получим число строф, разделив первую длину на вторую и добавив строфу, вычтенную из высоты текста:

```
\setcounter{verse}{1*\ratio{1\numenator}{1\denumenator} +1}
```

Данные действия учитывают специфику вычисления счетчиков и длин. Так как команда `\ratio` дает действительное число, ее результат умножается на единицу. Чтобы длины числителя и знаменателя заведомо были жесткими, они умножаются слева на единицу.

Полученное значение, `\theverse` ⇝ 7, показывает, что для поэмы из ста строф понадобится пятнадцать страниц, при этом пять строф остается в «запасе». На их месте можно разместить название поэмы и ее эпиграф.

Возможность использования в вычислениях параметров фрагмента текста, обеспечивают команды

```
\settowidth{команда}{текст},
\settodepth{команда}{текст}.
\settoheight{команда}{текст},
\settototalheight{команда}{текст}.
```

Они сохраняют ширину, глубину, высоту и полную высоту *текста* в длины, которые должны быть заранее объявлены декларациями `\newlength`. Сам текст не печатается. Три первые команды определены в ядре LaTeX, а последнюю добавляет пакет calc. Он также вводит ряд команд:

```
\widthof{текст},   \heightof{текст},
\depthof{текст},   \totalheightof{текст},
```

выполняющих две функции. Они печатают *текст* и при этом являются длинами с соответствующими размерами. В печатном виде текст может насчитывать несколько строк, но при вычислении размеров он представляется в виде единой длинной строки. Если эти команды находятся в аргументах команд `\setlength` или `\addtolength`, текст не печатается.

Для примера присвоим длине `\Length` значение длины `\widthof` и напечатаем его:

```
\setlength\Length{\widthof{Для примера ... и напечатаем его:}}
  \the\Length ⇝ 382.8601pt,      \the\linewidth ⇝ 321.51613pt.
```

Аргументом команды \widthof послужила предыдущая фраза, длина которой, как видно, превышает длину строки.

7.6. Подчеркивание

Ранее обсуждалась команда \underline, определенная в ядре LaTeX для подчеркивания строки или ее части. Дополнительные виды подчеркиваний обеспечивает пакет ulem [53], который вводит для них следующие команды:

\uline{текст}	подчеркивание,
\uuline{текст}	двойное подчеркивание,
\uwave{текст}	волнистое подчеркивание,
\dashuline{текст}	пунктирное подчеркивание,
\dotuline{текст}	подчеркивание точками,
\sout{текст}	~~вычеркивание~~,
\xout{текст}	*штриховка*.

В отличие от команды \underline они могут подчеркнуть несколько строк в пределах одного абзаца. Глубина, на которой рисуются линии, автоматически подстраивается под шрифт, которым набран текст, но, используя длину \ULdepth, ее можно зафиксировать, задав требуемую длину. Чтобы вернуть автоматический режим, ей нужно присвоить специальный флаг \maxdimen. Толщину линии регулирует команда \ULthickness{длина}. Ее можно изменять с помощью команды \renewcommand (см. разд. 7.1).

По умолчанию пакет ulem заменяет текст, выделенный командой \emph, подчеркнутым текстом. Декларация \normalem отменяет этот режим, а декларация \ULforem вновь его возвращает. Замену можно отменить глобально, загрузив пакет с параметром normalem.

Пакет ulem позволяет вводить новые подчеркивающие команды, устанавливать цвет линий и заменять подчеркиванием команды и декларации, изменяющие любые атрибуты шрифта, кроме кегля. Как это делать, описано в его документации.

7.7. Цветная верстка

Текст научной публикации раскрашивать не принято, но некоторые компоненты электронных документов можно выделять цветом.

Например, пакет hyperref использует разноцветные рамки и цифры для выделения гиперссылок. Как правило, иллюстрации делаются цветными и, если красные кружки, синие квадраты и зеленые треугольники использованы при построении кривых, вполне допустимо в подписи к рисунку указать: «Символами ●, ■ и ▲ помечены кривые...».[5]

Возможность цветной верстки предоставляет стандартный пакет color [54], имеющий следующие настройки:

dvipsnames — обеспечивает загрузку 68 предустановленных цветов, представленных на рис. 7.1;

usenames — позволяет использовать названия этих цветов в командах цветной верстки;

monochrome — отключает действие команд цветной верстки, позволяя оставить их в рукописи.

Следует учитывать, что пакет hyperref автоматически загружает пакет color без настроек, а так как повторная загрузка пакетов блокируется, пакет с настройками нужно загружать до него.

Для работы с цветом постпроцессору, генерирующему документ, требуется драйвер с описанием палитр. Компилятор pdflatex автоматически загружает драйвер pdftex для pdf-документов или dvips для PostScript-документов. При работе с другими компиляторами и постпроцессорами имя драйвера [3, 4] нужно указать вместе с настройками в параметре команды `\usepackage`, загружающей пакет color.

По умолчанию определены только восемь стандартных цветов:

black	(черный),	red	(красный),
yellow	(желтый),	green	(зеленый),
cyan	(голубой),	blue	(синий),
magenta	(пурпурный),	white	(белый).

Декларация `\definecolor{`*название*`}{`*палитра*`}{`*цвет*`}` позволяет смешать любой другой цвет в одной из палитр, т. е. моделей смешивания цветов, и использовать его по названию в командах раскраски.

Простейшей монохромной палитрой является `gray`. Оттенки серого цвета задаются действительным числом, изменяющимся от нуля до единицы, соответствующих черному и белому цвету.

Пример: `\definecolor{gray75}{gray}{0.75}` .

[5] При печати на монохромном принтере цвета нивелируются, поэтому формы символов должны различаться.

Рис. 7.1. Загружаемые цвета пакета color

Палитра `rgb` (red-green-blue) смешивает красный, зеленый и синий цвета. Значения компонент задает тройка чисел, изменяющихся от нуля до единицы, перечисляемых через запятую.

Пример: `\definecolor{MyGreen}{rgb}{0.15,0.75,0.15}` .

Набор `{1,1,1}` соответствует белому цвету.

Базой палитры `cmyk` (cyan-magenta-yellow-black) являются голубой, пурпурный, желтый и черный цвета. Цвет определяют четыре числа, также изменяющиеся от нуля до единицы и разделенные запятыми. Белому цвету соответствует набор `{0,0,0,0}`.

Пример: `\definecolor{LB}{cmyk}{0.5,0.25,0.15,0.05}` .

Цветная верстка предусматривает изменение цветов текста и фона. Цвет текста задает декларация `\color[`*палитра*`]{`*цвет*`}` , или команда `\textcolor[`*палитра*`]{`*цвет*`}{`*текст*`}` . Цвет можно синтезировать непосредственно при «раскрашивании», указав палитру в параметре команд и набор смешиваемых цветов в их аргументе. Можно также использовать название одного из уже созданных или загруженных цветов, указав палитру `named`, которую можно опустить, если пакет color загружен с параметром usenames.

Приведем примеры, совместив их код с результатом верстки:

```
\textcolor[rgb]{0.15, 0.6, 0.15}{\bf цвет палитры rgb}.
    {\color{blue}\bf стандартный цвет}.
```

В упоминавшейся подписи к рисунку цветные символы можно сделать разными способами, так как в формулах команды `\color` и `\textcolor` тоже работают:

```
\textcolor{red}{$\bullet$}      ⇝  ●,
$\textcolor{blue}\blacksquare$  ⇝  ■,
$\color{green}\blacktriangle$   ⇝  ▲.
```

При использовании параметра dvipsnames пакет color загружает цвета, показанные на рис. 7.1. Они определены в палитре `named` декларациями `\DefineNamedColor{named}{`*название*`}{`*палитра*`}{`*цвет*`}` . Как уже говорилось, если пакет загружен без параметра usenames, палитру нужно указывать, используя цвет:

```
\textcolor[named]{Plum}{\bf цвет Plum палитры named}.
```

Загрузка с параметром usenames позволяет пользоваться цветом без указания палитры: `{\bf\color{Magenta}` **пурпурный цвет**`}`.

Декларация `\definecolor{`*название*`}{named}{`*цвет*`}` вводит синоним цвета палитры `named`. Такой цвет используют обычным образом:

```
\definecolor{DarkVinous}{named}{Maroon}
\textcolor{DarkVinous}{\bf темно-бордовый цвет}.
```

В дополнение к цвету текста в боксах можно задать фон. Его цвет устанавливают команды

```
\colorbox[палитра]{цвет фона}{текст} ,
\fcolorbox[палитра]{цвет рамки}{цвет фона}{текст} .
```

Вторая из них создает бокс в рамке, при этом цвета фона и рамки синтезируются в общей палитре. Цвет задается так же, как в командах `\color` и `\textcolor`. Бокс формируется в виде одной строки текста, но это ограничение легко обойти, поместив в него окружение minipage или команду `\parbox`.

Покажем, как в этой книге задавался фон команд и окружений, названий пакетов и т. д., к которым адресуется предметный указатель:

```
\definecolor{Bckgd}{rgb}{0.95,0.95,0.95}
\colorbox{Bckgd}{серый фон} ⇝ серый фон .
```

По умолчанию фон страниц прозрачен. Его можно установить декларацией `\pagecolor[палитра]{цвет фона}` или вновь убрать декларацией `\nopagecolor`. Он изменяется, начиная со страницы, на которой находится начало абзаца, содержащего одну из этих деклараций, а если на странице их окажется несколько, фон задаст последняя. Область действия этих деклараций не ограничивается группой, поэтому с их помощью не удастся залить фоном лишь часть страницы.

Богатые возможности цветной верстки дает пакет xcolor [55]. Многие пакеты загружают его неявно с этой целью. Девятнадцать цветов он вводит по умолчанию:

white	(белый).		
black	(черный),	lime	(лайм),
darkgray	(темно серый),	green	(зеленый),
gray	(серый),	olive	(оливковый),
lightgray	(светло серый),	teal	(зелено-голубой),
pink	(розовый),	cyan	(голубой),
brown	(коричневый),	blue	(синий),
red	(красный),	violet	(фиолетовый),
orange	(оранжевый),	purple	(пурпурный),
yellow	(желтый),	magenta	(светло пурпурный).

В добавок к ним пакет предлагает загрузить еще три палитры с именованными цветами: палитра dvipsnames содержит 68 цветов, палитра

svgnames располагает 151 цветом, а в палитре x11names определено 318 цветов, при этом в разных палитрах цвета с одинаковыми названиями имеют разные оттенки. В двенадцати палитрах пользователь может смешать собственные цвета. Есть возможность определять непрерывные последовательности цветов, как показано на рис. 7.2 на примере спектра оптического излучения.

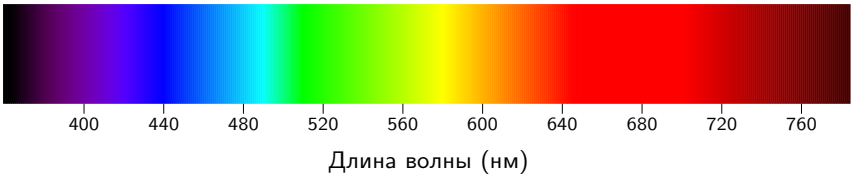


Рис. 7.2. Спектр оптического излучения

При работе с пакетом xcolor действуют все описанные ранее команды. Он расширяет их синтаксис, добавляет им новые возможности, а также вводит ряд новых команд. Будучи загружен с параметром table, он поддерживает цветную верстку таблиц, описанную в разд. 10.6, автоматически загружая для этого пакет colortbl. Описание всех ресурсов и настроек пакета xcolor заинтересованный читатель найдет в его документации [55].

7.8. Гиперссылки

В электронном документе ссылки на литературу, формулы, рисунки и т. д. удобно преобразовать в гиперссылки, позволяющие легко перейти к объекту ссылки. Такую возможность предоставляет пакет hyperref. Он также способен настроить практически все аспекты, связанные с представлением pdf-документа, вплоть до деталей оформления окна открывающей его программы и корректировки пунктов ее меню. Пакет имеет чрезвычайно большое число настроек, вводит новые команды и переопределяет большое количество стандартных, поэтому его рекомендуется загружать последним. Если он добавляется в рукопись, которая уже компилировалась, рекомендуется удалить все промежуточные файлы, иначе первые два прохода компиляции могут заканчиваться ошибками. То же нужно сделать и после его отключения. Не обсуждая богатые ресурсы пакета hyperref, описанные в его документации [56] и

книгах [3, 5], упомянем лишь минимальный набор настроек и команд, относящихся непосредственно к гиперссылкам.

При загрузке пакета hyperref ссылки, генерируемые командами `\ref{•}`, `\pageref{•}`, `\eqref{•}` и `\cite{•}`, автоматически становятся гиперссылками, а чтобы сделать обычную ссылку введены команды

`\ref*{`*метка*`}`, `\pageref*{`*метка*`}` и `\autoref{`*метка*`}`.

Последняя в зависимости от типа объекта, на который ссылаются, добавляет перед номером контекстную вставку, например номер формулы `\autoref` выведет в виде «Equation •» в английском тексте и «выр. •» в русском. Контекстные вставки хранятся в командах, перечисленных в табл. 7.3. С помощью переопределения `\renewcommand` их можно изменить.

Таблица 7.3

Контекстные вставки команды `\autoref`

Команды	Контекстные вставки	
`\AMSautorefname`	Equation	выр.
`\equationautorefname`	Equation	выр.
`\theoremautorefname`	Theorem	теор.
`\Hfootnoteautorefname`	footnote	подстр. прим.
`\figureautorefname`	Figure	рис.
`\Itemautorefname`	item	п.
`\pageautorefname`	page	с.
`\tableautorefname`	Table	табл.
`\appendixautorefname`	Appendix	прил.
`\chapterautorefname`	chapter	гл.
`\paragraphautorefname`	paragraph	п.
`\partautorefname`	Part	ч.
`\sectionautorefname`	section	разд.
`\subsectionautorefname`	subsection	разд.
`\subsubsectionautorefname`	subsubsection	разд.

Вдобавок к перечисленным ссылкам определена команда

`\hyperref{`*имя метки*`}{`*текст*`}`.

Ее первым аргументом является имя метки, заданной командой `\label`, а во второй аргумент помещается текст, оформляемый в виде гиперссылки. Этим она отличается от команд `\ref`, `\pageref`, `\eqref` и `\cite`, у которых гиперссылками становятся выводимые номера или ярлыки.

Для создания гиперссылок пакет hyperref использует упоминавшуюся в разд. 7.4 команду `\refstepcounter{•}`, связывающую метки со значениями счетчиков. Но так как в нумерации страниц данная команда не участвует, гиперссылки, формируемые командами `\pageref`, могут оказаться неверными, хотя номера страниц будут напечатаны правильно. Чтобы этого не происходило, пакет hyperref вводит специальный счетчик и команду `\phantomsection`, связывающую его с метками. Ее нужно ставить перед командами `\label`, чтобы пометить страницы, на которые предполагается ссылаться, например:

```
\phantomsection\label{p:sample}  ...  \pageref{p:sample}.
```

Эту команду можно использовать также, если гиперссылка элемента оглавления, вносимого в вручную, открывает неверную страницу:

```
\phantomsection
\addcontentsline{toc}{chapter}{Указатель фамилий}.
```

Команда `\hyperref` адресуется к объектам, находящимся в самом документе, а чтобы делать гиперссылки к внешним ресурсам, определена команда `\href{`*ресурс*`}{`*текст*`}`. Ее второй аргумент также задает текст, оформляемый в виде гиперссылки. *Ресурс* может быть интернет-адресом, адресом электронной почты или путем к файлу. Переход по гиперссылке ведет к выполнению различных действий, зависящих от идентификатора ресурса.

Адрес почты должен начинаться с идентификатора `mailto:`. Например переход по ссылке `e-mail`, созданной командой

```
\href{mailto:anybody@gmail.com}{e-mail},
```

откроет почтовую программу для редактирования письма с заготовленным адресом `anybody@gmail.com`.

Путь к файлу должен начинаться либо с идентификатора `file://` или `run:`, либо со слэша или точки.[6] Если идентификатор отсутствует, автоматически используется `file://`. Например, команда

```
\href{run:sample.avi}{видео}
```

создаст гиперссылку `видео` для запуска медиафайла, находящегося в той же папке что и pdf-документ. Команда

```
\href{d:/Photos/sample.jpg}{фотография}
```

[6] Путь к файлу указывается в Unix-нотациях, т. е. папки разделяются символом `/`, в относительном пути рабочая папка обозначается точкой, а родительская — двумя точками. В операционной системе Windows абсолютный путь начинается с идентификатора диска, например `c:/Program Files/...`

привяжет гиперссылку `фотография` к снимку `sample.jpg` в папке `Photos` на диске `d:`.

Результат действий, совершаемых при переходе по гиперссылке, зависит от типа файла, настроек программы, показывающей pdf-документ, и настроек безопасности операционной системы. При правильных настройках файл должен открываться программой, ассоциированной в операционной системе с его расширением.

Интернет-адрес обычно начинается с идентификатора типа ресурса `http://`, `https://`, `ftp://` и т. п. Например:

```
\href{https://en.wikibooks.org/wiki/LaTeX}{LaTeX}.
```

Переход по гиперссылке запускает браузер, загружающий нужную интернет-страницу. Если идентификатор не указан, автоматически используется `http://`.

Для ссылок на интернет-ресурсы определены команды

`\url{`*интернет-ресурс*`}` и `\nolinkurl{`*интернет-адрес*`}`.

Аргумент первой используется и как адрес, и как текст гиперссылки, а вторая вставляет в документ только адрес, не создавая гиперссылку.

Рассмотрим теперь вкратце настройку пакета hyperref. Ее можно сделать при загрузке пакета, перечислив настройки списком через запятую в параметре команды `\usepackage`, например

```
\usepackage[colorlinks,bookmarks,unicode]{hyperref}.
```

Многие настройки имеют значение true (используемый), false (неиспользуемый) или присваивают цвет. Значение можно установить путем присваивания, например `hyperindex = false`, а если параметр указан без присваивания, используется true.

Обычно текст гиперссылки заключается в цветную рамку или сам выделяется цветом. Формат присваивания цвета зависит от оформления гиперссылки. Для рамки указываются значения компонент палитры rgb, разделенные пробелом, а для текста — имя цвета, например

```
citebordercolor = .5 0 1, citecolor = blue.
```

Задание цвета текста палитрой не поддерживается, поэтому имена нестандартных цветов нужно определять заранее. Отметим, что пакет xcolor, загруженный с параметром hyperref позволяет задать цвета рамок и текста множеством более удобных способов.

Перечислим основные настройки пакета hyperref и их назначение.

bookmarks — формирует электронное оглавление, обычно выводимое слева от текста в окне программ просмотра pdf-документов.

unicode — обеспечивает unicode-кодировку pdf-конструкций, в частности электронного оглавления.

hyperindex — формирует гиперссылки предметного указателя.

linktocpage — оформляет в виде гиперссылок номера страниц в оглавлении. По умолчанию гиперссылками служат названия разделов.

hidelinks — убирает раскраску элементов и рамок, сохраняя гиперссылки. Применяется для печати документов.

colorlinks — выделяет цветом текст гиперссылки. По умолчанию вокруг гиперссылки рисуется тонкая цветная рамка, а ее текст не раскрашивается.

pdfborder — задает толщину рамок, окружающих гиперссылки. По умолчанию ее значение равно 0 0 1, где единица означает 1pt. Параметр colorlinks сбрасывает значение в 0 0 0.

anchcolor — задает цвет текста, служащего объектом гиперссылки. По умолчанию используется черный (black). Данную настройку pdflatex не поддерживает.

allcolors , allbordercolors — задают цвета всех рамок и всех гиперссылок. По умолчанию не используются.

citecolor , citebordercolor — задают цвета гиперссылок на библиографию. По умолчанию используется зеленый (green, 0 1 0).

filecolor , filebordercolor — задают цвета гиперссылок на файлы. По умолчанию используется голубой (cyan, 0 .5 .5).

linkcolor , linkbordercolor — задают цвета гиперссылок на сноски, пункты оглавления, страницы указателя и объекты, помеченные командами `\label`. По умолчанию используется красный (red, 1 0 0).

menucolor — задает цвет пунктов меню программы просмотра pdf-документов. По умолчанию используется красный (red, 1 0 0).

runcolor , runbordercolor — задают цвета гиперссылок с идентификатором `run:`, запускающих внешние приложения. По умолчанию для рамки используется цвет 0 .7 .7, а для текста — цвет filecolor.

urlcolor , urlbordercolor — задают цвета гиперссылок на интернет-ресурсы. По умолчанию используется пурпурный (magenta, 0 1 1).

Отметим, что параметр unicode необходим для корректной русификации электронного оглавления при восьмибитной кодировке манускрипта.

Действия пакета hyperref можно настроить и после загрузки. Для этого определена команда `\hypersetup{`*список настроек*`}`, в аргументе которой через запятую перечисляется *список настроек*:

```
\hypersetup{ bookmarks=true, unicode=true,
             colorlinks=true, citecolor=blue, urlcolor=cyan }.
```

С ее помощью большинство настроек можно изменять или добавлять многократно в любом месте рукописи, однако часть из них устанавливается только в преамбуле. Например, вне преамбулы изменить стиль оформления гиперссылок, задаваемый параметром colorlinks, не удастся, зато корректировать их цвета можно в любом месте.

В заключение добавим, что hyperref позволяет заполнить в pfd-форме с информацией о документе поля pdftitile, pdfauthor, pdfsubject, pdfkeywords, pdfcreator, pdfproducer. В последние два поля автоматически заносятся название компилятора и информация об использованном программном обеспечении. Например, в данной книге это

```
pdfproducer={LuaTeX-1.17.0},
pdfcreator={LaTeX with hyperref}.
```

Остальные поля авторы могут заполнить сами:

```
\hypersetup{ pdftitle = {Основы LaTeX}, pdfauthor = {А.В.Кузнецов},
          pdfsubject = {учебное пособие}, pdfkeywords = {LaTeX} }.
```

7.9. Конвертация графических файлов

В разд. 3.1 говорилось, что программа epstopdf конвертирует eps-рисунок в формат pdf. Стандартный пакет epstopdf позволяет делать это непосредственно во время компиляции. Для этого также нужно загрузить пакет graphicx с параметром pdftex, а чтобы иметь возможность запускать конвертирующую программу, в командную строку компиляторов LaTeX нужно добавить ключ `-shell-escape` в ОС Linux, или `-enable-write18` для ОС Windows.

Когда все условия выполнены, для файлов с расширением •.eps команды `\includegraphics` ищут соответствующие pdf-файлы, а если их нет, вызывают программу epstopdf и затем загружают конвертированные файлы.

Действия пакета epstopdf контролируют параметры, которые можно задать как при его загрузке, так и после нее. Для настройки предназначена команда

`\epstopdfsetup{`*список параметров*`}`,

в аргументе которой через запятую перечисляются параметры. Приведем лишь те из них, что используются наиболее часто.

update — конвертировать файл только, если pdf-файл не существует, или создан ранее eps-файла.

suffix= • — добавить суффикс • к имени pdf-файла. При установке параметра suffix= -generated файл `sample.eps` конвертируется в `sample-generated.pdf`.

prefersuffix= **true** | false — вставлять в документ файлы с суффиксом (true) или без него (false).

Последний параметр определяет выбор файлов, если в рабочей папке находятся два сорта pdf-файлов, среди которых одни имеют те же имена, что и исходные eps-файлы, а имена вторых содержат суффикс, указанный параметром suffix.

Параметры update, suffix= -eps-converted-to, prefersuffix= true заданы по умолчанию, поэтому автоматически генерируются и загружаются файлы с именами •`-eps-converted-to.pdf`.

По умолчанию файлы конвертирует программа epstopdf, но настройки позволяют использовать и другие программы. Также можно настроить конвертацию файлов с форматами, отличными от eps [57].

Глава 8

Настройка документа

В первой части книги описана верстка небольших документов с простой структурой типа статьи. Работа с большими текстами имеет свою специфику. Например книги, как правило, имеют указатели, библиография в них может формироваться по главам, а страницы могут иметь верхние колонтитулы. Их создание и настройка, а также другие вопросы, которых мы еще не касались, разбираются далее.

8.1. Общие настройки

Начнем с оптимизации верстки, осуществляемой пакетом `microtype` [58]. Настраивая параметры верстки на самом низком уровне, он не вносит в рукопись никаких команд. Если стандартные алгоритмы TEX регулируют заполненность строк, распределяя в них слова, то пакет microtype идет дальше, он слегка меняет ширину букв в словах, корректируя символы шрифтов. Для таких операций лучше всего подходит pdf-формат документа и векторные шрифты, например упоминавшиеся в разд. 1.1 PostScript-шрифты пакета `cm-super` или OpenType-шрифты.

Хотя пакет microtype имеет параметры, его можно загружать и без них. Его настройки по умолчанию хорошо отлажены, и их лучше не трогать. Загрузив пакет, с помощью команды

```
\UseMicrotypeSet[операции]{шрифты}
```

следует выбрать один из готовых наборов *операций* и *шрифтов*. Наиболее полным является набор

```
\UseMicrotypeSet[protrusion]{basicmath},
```

оптимизирующий расстояния между словами и ширины букв всех текстовых шрифтов нормальной насыщенности, исключая моноширинные, в диапазоне размеров от `\footnotesize` до `\large` (см. табл. 2.7). Вдобавок к этому оптимизируется ширина букв математических шрифтов с кодировками OML и OMS [3].

Пакет microtype вводит и другие команды, позволяющие настраивать различные операции и формировать свои собственные наборы операций и шрифтов, но лучше довериться профессионалам, разработавшим настройки, используемые по умолчанию. Шрифты — очень тонкий инструмент, к которому нужно относиться с большим уважением и осторожностью.

LaTeX управляет межстрочным интервалом с помощью длины `\baselineskip` и коэффициента `\baselinestretch`. Оптимизируя эту операцию, пакет setspace [59] определяет декларации `\singlespacing`, `\onehalfspacing` и `\doublespacing`, активирующие одинарный, полуторный или двойной интервалы. Например, поставленная в преамбуле рукописи декларация `\onehalfspacing` установит для всего документа полуторный интервал. Минимальной единицей текста, на который действуют декларации, является абзац. Интервал не изменится, если использовать их в группе, не имеющей пустой строки или команды `\par`. Кроме деклараций, пакет setspace вводит окружения singlespace, onehalfspace и doublespace, изменяющие интервал в части текста. Две последние предназначены для *увеличения* интервала, поэтому окружение onehalfspace не изменит его, если ранее задан двойной интервал. Декларация `\setstretch{`*число*`}` и окружение space,

```
\begin{space}{число},
```

позволяют установить произвольную величину межстрочного интервала, где число — значение коэффициента `\baselinestretch`.

Оформление заголовков разделов определяет класс документа. Пакет titlesec [60] дает возможность изменить его по своему усмотрению. Для настройки печати номеров разделов он вводит декларацию `\titlelabel{`*формат номера*`}`. Команды, указанные в ее аргументе, применяются к разделам всех уровней. В ней можно задать отбивки номеров и шрифт для их печати. Сами номера следует задать командой `\thetitle`. При печати ее автоматически заменят команды, выводящие номера разделов, например `\thesection` для `\section` или `\theparagraph` для `\paragraph`. Если задать

```
\titlelabel{\sf\quad\thetitle.\quad},
```

номера будут завершаться точкой и печататься рубленым шрифтом, окруженные пробелами шириной 1em.

Декларация `\titleformat*{`*команда*`}{`*формат печати*`}` позволяет настроить печать заголовков разделов уровня, указанного командой. Например, так можно задать курсивные заголовки параграфов: `\titleformat*\paragraph{\itshape}`.

Еще одна декларация регулирует отбивку заголовков:

`\titlespacing{`*команда*`}{`*отступ*`}{`*сверху*`}{`*снизу*`}` .

Например, настройка `\titlespacing\section{3em}{2\baselineskip}{0pt}` задаст для разделов `\section` левый отступ шириной 3em, отобьет заголовок от предыдущего текста на два межстрочных интервала и уберет зазор между ним и последующим тестом. Вариант «со звездочкой» этой декларации убирает отступ абзаца, следующего за заголовком.

Детализированную настройку заголовков выполняет декларация

`\titleformat{`*команда*`}[`*тип*`]{`*заголовок*`}{`*номер*`}{`*зазор*`}{`*до*`}{`*после*`}` .

Как и в предыдущих случаях, ее первый аргумент определяет уровень раздела. С помощью параметра можно выбрать один из предустановленных *типов* вывода заголовков. Аргументы, следующие за параметром, задают форматы печати *заголовка* и *номера* и ширину *зазора* между ними.[1] В последних двух аргументах можно указать код, который будет исполняться *до* и *после* вывода заголовка.

Мы рассмотрели лишь небольшую часть ресурсов пакета titlesec, предоставляющего доступ ко всем аспектам верстки заголовков. Более подробную информацию заинтересованный читатель найдет в его документации [60].

8.2. Большие документы

Большой документ удобно редактировать по частям, разбив его на несколько файлов, сводимых воедино при компиляции. Для этого нужно создать основной файл программы, содержащий преамбулу, и добавить в него команды загрузки других файлов.

Существует два варианта загрузки. При загрузке файла командой `\input{`*имя файла*`}` его текст становится непосредственным продолжением предшествующего текста, а команда `\include{`*имя файла*`}` ,

[1] Этот параметр является длиной, например 0.5em.

вставляя файл, начинает новую страницу. В файлах, загружаемых командой `\include`, можно использовать загрузки `\input`, но вложенные загрузки `\include` запрещены. Подгружаемые файлы, в свою очередь, могут содержать команды `\input`, причем уровень вложенности таких загрузок не ограничен. Команды `\include` должны находиться в тексте программы *после* команды `\begin{document}`. Команды `\input` могут использоваться в любом месте, но файлы, загружаемые в преамбуле, должны содержать только код. Некоторые пакеты загружают так свои дополнительные ресурсы.

Если поместить в файл команду `\endinput`, загрузится лишь предшествующая ей часть файла. В основном файле рукописи данную команду использовать нельзя, так как компиляция завершится без создания документа.

Можно компилировать лишь часть документа. Для этого его следует разбить на файлы, загружаемые командами `\include`, а затем в команде `\includeonly{`*список файлов*`}` перечислить только нужные. Данная команда должна стоять в преамбуле рукописи *до* команды `\begin{document}`.

В командах `\include` и `\input` имена файлов указываются по одному, а в `\includeonly` — списком через запятую. Файлы, загружаемые командами `\include`, должны иметь расширение •.tex, но при этом в командах `\include` и `\includeonly` их имена указываются без расширения. Команды `\input` загружают файлы с любым расширением, а если оно опущено, автоматически используется •.tex.

Команда `\includeonly` не влияет на загрузки `\input`. Отменить последние удастся, только отправив эти команды в комментарий. При этом потеряются данные о нумерации страниц и объектов, содержащихся в незагруженных файлах, в результате чего общая нумерация может сбиться. Для файлов, загружаемых командой `\include`, генерируются собственные aux-файлы, позволяющие поддерживать правильную нумерацию, даже если сами файлы не загружаются. В файлах, загружаемых командой `\include`, нельзя создавать новые счетчики, декларации `\newcounter{•}[•]` в них не работают.

Подведем итог сказанному выше. Чтобы эффективно использовать возможность компиляции частей документа, разумно разбить его на главы, загружаемые командами `\include`, а большие главы дополнительно разбить на файлы, подгружаемые командами `\input`. Такое разбиение обеспечит правильную нумерацию страниц и ссылок до-

кумента в целом, а использование `\includeonly` позволит избежать загрузку лишних файлов при редактировании отдельных глав.

8.3. Библиография к разделам

В некоторых книгах для каждой главы формируется собственный список литературы. В LaTeX это делает пакет chapterbib в сочетании с генерацией библиографии компилятором BibTeX. Чтобы сверстать литературу таким образом, рукопись нужно разбить на главы, загружаемые командами `\include`, при этом каждая из глав должна содержать команды `\bibliographystyle` и `\bibliography`, при отсутствии которых цитирование окажется невозможным, так как chapterbib ограничивает поиск источников только загружаемым разделом. Автоматически библиография генерируется только для основной части рукописи, содержащей команды `\include`, а чтобы сформировать ее к главам, BibTeX придется запустить вручную с •.aux-файлом каждой главы. Таким образом, для формирования документа в целом нужно запустить общую компиляцию, затем сгенерировать библиографии глав и вновь повторить общую компиляцию.

По умолчанию для документов класса article библиография оформляется в виде разделов `section*`, а для книг (book) или отчетов (report) как ненумерованные главы `chapter*`. В последнем случае уровень библиографии можно понизить до `section*`, загрузив пакет с параметром sectionbib. Тогда литература основной части будет сверстана в виде главы `chapter*`, а загружаемых глав — в виде разделов `section*`.

В документации пакета chapterbib [61] описаны команды, позволяющие формировать литературу и без выделения глав в отдельные файлы, но их использование делает документ нестандартным, поэтому мы их рассматривать не будем. Исключение сделаем только для декларации `\CitationPrefix{`*текст*`}`, добавляющей *текст* перед номером источника в библиографиях и ссылках. Если в преамбуле рукописи задать `\CitationPrefix{\thechapter.}`, номера ссылок будут предворяться номером главы и точкой.

8.4. Стили цитирования

В первой части книги обсуждалась верстка списка литературы, цитирование источников, создание и редактирование библиографических

баз. Здесь мы опишем оформление ссылок с помощью пакетов cite и natbib и разберем особенности стиля цитирования автор-год.

8.4.1. Оформление ссылок

Пакет cite [39] предназначен для работы с нумерованными списками литературы, взаимодействуя с которыми, команда `\cite` по умолчанию заключает ссылки в квадратные скобки и выводит их в строку как обычный текст. Пакет cite настраивает стиль оформления ссылок, генерируемых этой командой. Основная настройка осуществляется при его загрузке с помощью следующих параметров:

sort | nosort — (не) сортировать номера источников в ссылках;

compress | nocompress — (не) заменять последовательные номера диапазоном;

space | nospace — заменить тонкий пробел, разделяющий номера в ссылках, на обычный, или убрать его;

adjust | noadjust — (не) регулировать пробел перед ссылками;

nobreak — запретить разрывы строк в ссылках;

superscript , super — оформить ссылки в виде верхних индексов;

move | nomove — (не) переносить вперед знаки препинания, поставленные после ссылок, оформленных в виде верхних индексов;

ref — добавить слово «Ref.» перед номером в ссылках, имеющих уточняющую информацию, т. е. оформленных с помощью команды `\cite[•]{•}`;

biblabel — привести формат вывода номера источника в списке литературы в соответствие с форматом вывода ссылок командой `\cite`.

По умолчанию списки номеров в ссылках сортируются, сокращаются до диапазонов и выводятся в квадратных скобках в строку, при этом команда `\cite` оптимизирует пробелы, стоящие перед ссылками. Такое оформление обеспечивает список параметров, выделенных жирным шрифтом.

Параметры superscript или super инициируют вывод ссылок в виде верхних индексов без скобок, пробелы перед ссылками игнорируются, а следующие за ними знаки препинания переносятся вперед, например: `Arseneau\cite{Arseneau-cite}.` ⇝ Arseneau.[39]

Для печати каждого элемента ссылки пакет cite вводит собственную команду. Команды `\citeleft` и `\citeright` выводят скобки, охватывающие ссылку, поэтому их переопределение декларациями `\renewcommand\citeleft(` и `\renewcommand\citeright)` заменит квадратные скобки круглыми: `\cite{Arseneau-cite}` ⇝ (39).

Команда `\citeform{`*номер*`}` печатает номера ссылок. В нее можно добавить команды, изменяющие шрифт и другие параметры печати. Так, после определения `\renewcommand\citeform[1]{\bf \color{red} #1}` номера станут красными и жирными: `\cite{Arseneau-cite}` ⇝ [39]. Отметим, что декларации изменяющие вывод номера, не влияют на оформление других элементов ссылок.

Команды `\citepunct`, `\citedash` разделяют элементы списков номеров в ссылках, а команда `\citemid` отделяет от них уточняющую информацию. По умолчанию диапазон номеров разделяет короткое тире `\endash`, их список — запятая и небольшой пробел, а уточняющая информация отделяется от номера запятой и обычным пробелом. Пробелы обеспечивают возможность автоматического разбиения длинных ссылок между строками. Определение `\renewcommand\citepunct{,}`, сохранив запятую, уберет пробелы. В результате списки номеров станут более компактными, но при этом еще и «монолитными». Это затруднит их встраивание в текст.

Команда `\OverciteFont` устанавливает шрифт ссылок, оформляемых в виде верхних индексов. Команда `\CiteMoveChars` хранит список знаков препинания, которые переносятся вперед, если стоят после таких ссылок. По умолчанию в него входят точка, запятая, двоеточие и точка с запятой.

Команда `\citen{`*список ярлыков*`}` выводит номера ссылок без скобок: `\citen{Arseneau-cite}` ⇝ 39. Обычно ее используют, чтобы напечатать ссылку в строке, когда основные ссылки оформляются в виде верхних индексов, но возможна и обратная ситуация:

`Arseneau.$^\text{\citen{Arseneau-cite}}$` ⇝ Arseneau.[39]

Для команды `\citen` в целях совместимости с пакетом natbib определен синоним — команда `\citenum{`*список ярлыков*`}`. Пользоваться ей более предпочтительно, так как natbib команду `\citen` не поддерживает.

8.4.2. Стиль цитирования автор-год

Для верстки нумерованной библиографии вполне достаточно ресурсов, предоставляемых стандартными стилями BibTEX и пакетом

cite. Работу со списками литературы, упорядоченными по фамилиям авторов, наиболее эффективно организует пакет natbib [62], поддерживающий в полном объеме также и нумерованные списки. Ввиду своей универсальности он используется многими издательствами, поэтому опишем его основные возможности, ограничившись лишь теми, что связаны с цитированием источников и потому важны для авторов.

Как правило, пакет natbib используется вместе со специальными стилями BibTEX, позволяющими реализовать все его возможности. Вместе с ним поставляются стили plainnat, abbrvnat и unsrtnat, выполняющие те же функции, что и стандартные стили, описанные в разд. 5.2.2. Для оформления библиографии по ГОСТ 7.0.5-2008 пакет gost, рассмотренный там же, добавляет к ним аналоги gost2008n, ugost2008n, gost2008ns и ugost2008ns. Обширный набор стилей aapmrev4-2, aipauth4-2, aipnum4-2, apsrev4-2, apsrmp4-2, elsarticle-harv, elsarticle-num, elsarticle-num-names поставляют коллекции revtex (класс revtex4-2) и elsarticle, в которых natbib является основным средством верстки библиографии.

Отличительной особенностью пакета natbib является формирование преамбулы окружения thebibliography с определениями команд, форматирующих списки литературы. Набор команд зависит от стиля верстки библиографии.

Покажем, как будут выглядеть список литературы и ссылка на источник при использовании natbib со стилем ugost2008n:

В рукописи:

```
... в книге \cite{Knuth-TeXbook-1993} описаны ...

\begin{thebibliography}{99}
\def\selectlanguageifdefined#1{
  \expandafter\ifx\csname date#1\endcsname\relax
  \else\selectlanguage{#1}\fi}
\providecommand*{\BibEmph}[1]{#1}
\providecommand*{\BibDash}{
  \ifdim\lastskip>0pt\unskip\nobreak\hskip.2em plus 0.1em\fi
  \cyrdash\hskip.2em plus 0.1em\ignorespaces}

\bibitem[Кнут (1993)]{Knuth93}
  \selectlanguageifdefined{russian}
  \BibEmph{Кнут~Дональд~Е.} Все про {\upshape\TeX}. \BibDash
  \newblock Протвино~: АО RD\TeX, 1993. \BibDash
  \newblock С.~575. \BibDash  \newblock ISBN:~5-900614-01-8.
```

В документе:

... в книге Кнут (1993) описаны ...

Список литературы

[1] Кнут Дональд Е. *Все про TEX* — Протвино: АО RDTEX, 1993 — С. 575. — ISBN: 5-900614-01-8

В примере показана только часть команд преамбулы. Видно, что по сравнению с кодом, генерируемым обычными стилями BibTEX, описания источников выглядят гораздо сложнее, но это позволяет очень гибко настроить формат вывода ссылок на них. Стиль оформления ссылок задают многочисленные параметры пакета natbib, из которых наиболее часто используются:

authoryear | numbers — стиль цитирования автор-год, или номер ссылки;

round | square | curle | angle — оформление ссылок в круглых, квадратных, фигурных или угловых скобках;

comma | **semicolon** | colon — разделение списка ссылок запятой, точкой с запятой или двоеточием;

super — оформление нумерованных ссылок в виде верхних индексов;

sort&compress | sort | compress — сортировка ссылок в нумерованных списках и замена последовательности номеров диапазонами (второй и третий параметры выполняют только одну из указанных операций);

sectionbib — генерация библиографии по разделам с уровнем не ниже, чем глава (для этого также нужно загрузить пакет chapterbib).

Как и прежде, жирным шрифтом выделены параметры, используемые по умолчанию и не требующие загрузки.

Пакет natbib предоставляет несколько настроенных стилей печати ссылок, перечисленных в табл. 8.1. По умолчанию используется стиль natbib, но декларация `\citestyle{`*стиль цитирования*`}` позволяет выбрать любой другой в любом месте рукописи, причем это можно делать многократно. Стиль, используемый по умолчанию, можно подстроить под свои нужды, описав элементы оформления ссылок

в команде `\setcitestyle{`*стиль цитирования*`}`. Синтаксис такого описания подробно обсуждается в документации natbib [62].

Стили цитирования обеспечивают широкий выбор печати ссылок как на библиографию, сверстанную в виде нумерованного списка, так и на списки, упорядоченные по фамилиям авторов. Ссылки с фамилиями можно заключить в скобки полностью, как показано во второй колонке таблицы, или, оставив фамилии в тексте, указав в скобках год или номер публикации (см. третью колонку). Чтобы обеспечить разные варианты ссылок, пакет natbib вводит команды двух типов и связывает с одним из них стандартную команду `\cite`. Связь, обозначенная в табл. 8.1 фоном, зависит от выбранного стиля цитирования, например `\cite=\citep` для стилей plain, cospar и nature.

Таблица 8.1

Стили оформления ссылок пакета natbib

Стили	`\citep{•}`	`\citet{•}`
plain	[1, 2, 3]	Пух [1, 2], Иа [3]
cospar	/1, 2, 3/	Пух /1 2/, Иа /3/
nature	текст[1,2,3]	Пух[1,2], Иа[3]
plainnat	[Пух, 1, 2, Иа, 3]	Пух [1, 2], Иа [3]
kluwer	(Пух, 1991, 2002, Иа, 1913)	Пух (1991, 2002), Иа (1913)
dcu	(Пух; 1991, 2002; Иа, 1913)	Пух (1991, 2002); Иа (1913)
egu	(Пух, 1991, 2002; Иа, 1913)	Пух (1991, 2002); Иа (1913)
agu	[Пух, 1991, 2002; Иа, 1913]	Пух [1991, 2002]; Иа [1913]
	Загрузка пакета с параметром authoryear	
natbib, agms	[Пух, 1991, 2002, Иа, 1913]	Пух [1991, 2002], Иа [1913]
	Загрузка пакета с параметром numbers	
natbib, agms	[1, 2, 3]	Пух [1, 2], Иа [3]

Ссылки с фамилиями авторов и следующим за ними годом выхода или номером публикации, заключенным в скобки, генерируют команды:

```
\Citet[перед][после]{список ярлыков},
\citet[перед][после]{список ярлыков},
\Citet*[перед][после]{список ярлыков},
\citet*[перед][после]{список ярлыков}.
```

Те из них, что содержат звездочку в имени, печатают в ссылке список авторов, остальные ограничиваются фамилией первого автора. Команды `\Citet` и `\Citet*` печатают титульные приставки, начиная с заглавных букв. С помощью параметров *перед* и *после* основной части ссылки, заключенной в скобки, можно вставить дополнительную информацию. Если задан один параметр, информация добавляется в конец ссылки. Общие ссылки коллектива авторов собираются вместе. В списках литературы, содержащих несколько работ коллектива, вышедших в свет в одном году, публикации сортируются по названию и к году добавляются буквы a, b, c и т. д. Покажем, как выглядят ссылки в стиле `egu`, взяв за основу оригинальные примеры пакета natbib:

`\citet{jon90}`	Jones et al. (1990)
`\citet*{jon90}`	Jones, Baker, and Williams (1990)
`\citet{jon90, jam91}`	Jones et al. (1990); James (1991)
`\citet{jam90-1,jam90-2,jam91}`	James (1990a, 1990b, 1991)
`\citet[see][chap.2]{jam91}`	James et al. (see 1991, chap. 2)
`\citet[review][]{jon90}`	Jones et al. (review 1990)
`\citet{dRob98}`	della Robbia (1998)
`\Citet{dRob98}`	Della Robbia (1998)
`\Citet*{dRob98}`	Della Robbia and Williams (1998)

Еще один набор команд формирует ссылки, полностью заключенные в скобки:

`\Citep[•][•]{•}`, `\Citep*[•][•]{•}`,
`\citep[•][•]{•}`, `\citep*[•][•]{•}`.

`\citep{jon90}`	(Jones et al., 1990)
`\citep*{jon90}`	(Jones, Baker, and Williams, 1990)
`\citep{jon90, jam91}`	(Jones et al., 1990; James, 1991)
`\citep{jam90-1,jam90-2,jam91}`	(James, 1990a, 1990b, 1991)
`\citep[see][chap.2]{jam91}`	(see James et al., 1991, chap. 2)
`\citep[review by][]{jon90}`	(review by Jones et al., 1990)
`\citep{dRob98}`	(della Robbia, 1998)
`\Citep{dRob98}`	(Della Robbia, 1998)
`\Citep*{dRob98}`	(Della Robbia and Williams, 1998)

Их синтаксис, а также синтаксис ряда приведенных далее команд аналогичен командам `\citet`, поэтому мы заменили описание их аргументов и параметров символами •.

Все перечисленные выше команды имеют варианты с суффиксом `al`, верстающие ссылки без скобок:

```
\Citealp[•][•]{•},  \Citealp*[•][•]{•},
\citealp[•][•]{•},  \citealp*[•][•]{•},
\Citealt[•][•]{•},  \Citealt*[•][•]{•},
\citealt[•][•]{•},  \citealt*[•][•]{•},
```

`\citealt*{jon90}`	Jones, Baker, and Williams, 1990
`\citealp{jam90-1,jam90-2,jam91}`	James, 1990a, 1990b, 1991
`\citealp[see][(chap.2)]{jon91}`	see Jones et al., 1991, (chap. 2)
`\citealt[review by][]{jon90}`	review by Jones et al., 1990
`\Citealp{dRob98}`	Della Robbia, 1998

Параметры команд цитирования позволяют вставить текст до и после ссылок, но не между ними. Команда `\citetext{`*текст и ссылки*`}` дает возможность произвольно комбинировать текст и ссылки. Результат верстки она заключает в скобки, например:

`\citetext{see review by \citealt{jam91}}` ⇝ [see review by James 1991].

Элементы ссылок можно печатать по отдельности. Команды

```
\Citeauthor[•][•]{•},       \Citeauthor*[•][•]{•},
\citeauthor[•][•]{•},       \citeauthor*[•][•]{•},
\citefullauthor[•][•]{•},   \Citefullauthor[•][•]{•}
```

выводят списки авторов без скобок, при этом `\citefullauthor` является синонимом `\citeauthor*`. Год опубликования источника или его номер в списке литературы печатают команды

`\citeyear[•][•]{•}`, `\citenum[•][•]{•}`, `\citeyearpar[•][•]{•}`.

Последняя из них заключает год в скобки. Использование их параметров, вставляющих текст перед и после ссылок, не приведет к ошибке, однако может дать неожиданный результат.

Даже при цитировании списка литературы, отсортированной по принципу автор-год, команда `\citenum` печатает порядковый номер источника, а не год его опубликования.

`\citeyearpar{jam90-2}`	⇝	(1990b),
`\citenum{jam90-2}`	⇝	4.

При загрузке пакета natbib с параметром super или использовании стиля nature ссылки нумеруются и оформляются в виде верхних индексов, но `\citenum` напечатает номер источника в строке:

```
Основополагающая работа
Ландау~[\citenum{Landau-JETP-1953}]
и ряд последующих статей\cite{...}
```

Основополагающая работа Ландау [2] и ряд последующих статей[3–7]...

Командой `\defcitealias{`*ярлык*`}{`*псевдоним*`}` источнику списка литературы можно присвоить псевдоним и задействовать его в оформлении ссылок. Печатают псевдонимы команды

```
\citetalias[после]{список ярлыков},
\citepalias[перед][после]{список ярлыков}.
```

Первая выводит их без скобок, а вторая — в скобках. В скобки до и после псевдонимов можно добавить комментарии, тогда как псевдонимам, печатаемым без скобок, комментарий положен только в конце:

```
\defcitealias{Ford-model-2010}{ТУ}
\citetalias[ч.~1]{Ford-2010}                 ⇝  ТУ, ч. 1,
\citepalias[прим.~2 к][ч.~1]{Ford-20102}     ⇝  (прим. 2 к ТУ, ч. 1).
```

Отметим, что пакет hyperref превращает псевдонимы в гиперссылки.

Заключая обзор пакета natbib, сделаем два замечания. Во-первых, по умолчанию ссылки на библиографию, сверстанную стилями пакета gost, заключаются в круглые скобки и оформляются в стиле egu. Чтобы вывести их в квадратных скобках, задайте стиль цитирования plain. Во-вторых, без необходимости к специфическими командами пакета natbib прибегать не стоит. В большинстве случаев возможностей стандартной команды `\cite` вполне достаточно. Чтобы сделать ее полным аналогом команд `\citet` и `\citep`, natbib делает ее нестандартной, добавляя ей второй параметр, но сохраняет при этом и стандартную форму, в которой текст одного параметра выводится перед ссылкой. Если пользоваться только ей, документ останется полностью стандартным и копирование его частей в другие документы, компилируемые без пакета natbib, не вызовет проблем.

Мы рассмотрели лишь часть ресурсов пакета natbib, позволяющую настроить ключевые аспекты верстки библиографии и ссылок. Остальные его возможности, в основном, касаются оформления списков литературы. Выбрав подходящий стиль верстки библиографии, автор с ними не сталкивается. При необходимости со всеми ресурсами можно ознакомиться в документации natbib [62].

В последние годы активно разрабатывается пакет biblatex, способный заменить все предшествующие ресурсы верстки библиографии. Очень сложное устройство, грандиозное число новых типов записей и полей в библиографических базах, а также настроек и команд, формирующих библиографию и оформляющих ссылки, сдерживают издателей от его использования. Заинтересованных читателей мы отсылаем к его документации [63], насчитывающей более трехсот пятидесяти страниц!

8.5. Редактирование специальных списков

Ранее говорилось, что, собирая информацию о разделах, компилятор составляет оглавление и выводит его, если рукопись содержит команду `\tableofcontents`. Аналогичным образом он может сформировать списки рисунков и таблиц. Для этого используются команды `\listoffigures` и `\listoftables`, которые после компиляции замещаются соответствующими списками. В них включаются краткие названия рисунков и таблиц, приведенные в параметре команд `\caption`, или полные названия, если параметр отсутствует.

Списки рисунков и таблиц формируются в файлах с расширениями •.lof и •.lot. Эти файлы вместе с оглавлением, находящемся в файле с расширением •.toc, состоят из записей вида

`\contentsline{`*тип*`}{`*текст*`}{`*номер страницы*`}`,
`\contentsline{`*тип*`}{`*текст*`}{`*номер страницы*`}{`*гиперссылка*`}`,

в которых *текстом* является название раздела или подпись рисунка или таблицы. Третий аргумент очевиден. Четвертый аргумент, используемый для формирования гиперссылок в pdf-документах, вводит пакет hyperref, описанный в разд. 7.8. *Типом* является одно из ключевых слов figure, table или имя команды, формирующей раздел. Приведем для ясности пример:

```
\contentsline{figure}{\numberline{2.1}\ignorespaces
                                Подпись рисунка}{37}{figure.caption.11}
```

Команда `\numberline{`*номер*`}` выводит номер рисунка (таблицы, раздела), а команда `\ignorespaces` удаляет следующие за ней пробелы.

Первый аргумент команды `\contentsline` определяет уровень записей в списке. В списках lot и lof все записи принадлежат верхнему уровню, но рисунки и таблицы разных глав разделяются небольшим вертикальным пробелом. В оглавлении уровни следуют иерархии разделов. К первому уровню принадлежат записи типа part и chapter, второй и третий составляют записи section и subsection. Разделы более низкого уровня по умолчанию в оглавление не включаются, но, как указано в разд. 7.4, это можно исправить, переопределив счетчик tocdepth.

В списки рисунков и таблиц информацию поставляют команды `\caption`, а в оглавление — команды, создающие разделы. Авторы могут корректировать параметры верстки списков и вносить в них дополнительную информацию. Команда `\addtocontents{`*список*`}{`*текст*`}` вставляет *текст* в *список*, указываемый аббревиатурой toc, lof или

lot. Ее второй аргумент обычно используют, чтобы вставить команды, настраивающие верстку списка, но с ее помощью можно добавить в него преамбулу и комментарии, состоящие из нескольких абзацев текста. В качестве примера покажем, как можно подавить создание новой страницы списка, когда на ней окажется единственная строка:

```
\addtocontents{•}{
    \protect\enlargethispage*{\baselineskip} \protect\nopagebreak}.
```

В данном случае команда `\protect` используется, чтобы следующая за ней команда не исполнялась, а заносилась в список. При необходимости перед длиной `\baselineskip` можно поставить увеличивающий или уменьшающий коэффициент. Если речь идет о списке таблиц или рисунков, команду `\addtocontents` можно разместить непосредственно в последнем плавающем объекте *до* последней команды `\caption`. В случае оглавления она должна стоять *перед* последним разделом, заносимым в оглавление.

Команду `\addcontentsline{`*список*`}{`*тип*`}{`*текст*`}` используют, чтобы сформировать запись вручную и добавить ее в список. Ее первый аргумент, как и в предыдущем случае, указывает *список*, в который направляется запись. *Тип* записи и ее *текст* становятся аргументами команды `\contentsline`, в которую компилятор добавит номер страницы и информацию для гиперссылки. Для примера покажем, как вставить в список lof рисунок, сделанный вручную (см. конец разд. 8.6):

```
\addcontentsline{lof}{figure}{\protect\numberline{\thefigure}%
\ignorespaces Подпись рисунка}.
```

Для правильной отбивки номер рисунка нужно поместить в команду `\numberline`, иначе оформление добавляемой строки будет отличаться от остальных пунктов списка. В данном случае следует защитить только команду `\numberline`, а команда `\thefigure` должна быть выполнена. Ее обязательно нужно поместить в скобки, чтобы составной номер рисунка попал в аргумент полностью.

В сложных случаях, после того как вся информация уже занесена в списки, их можно отладить вручную, однако при последующих компиляциях файлы автоматически обновятся и ручная правка будет утеряна. Команда `\nofiles` помогает избежать это. Ее присутствие в преамбуле рукописи отменяет генерацию служебных файлов, включая списки. Используйте ее, чтобы не потерять правку, а когда понадобится снова обновить списки, перед тем как убрать команду `\nofiles`, сделайте копии отредактированных файлов.

Пакет titletoc [60] позволяет настроить печать оглавлений. Такая задача возникает довольно редко, поэтому мы ограничимся лишь обсуждением команд, настраивающих вывод их элементов. В большинстве случаев для этого используют декларацию

```
\dottedcontents{уровень}[отступ]{формат}{номер}{отточие}
```

Ее первый аргумент задает уровень элемента, например section. Параметр устанавливает длину его отступа от края страницы. Следующий аргумент может содержать код, настраивающий печать элемента, например параметры шрифта и верхнюю отбивку. Третий аргумент задает ширину поля, отводимого под номер заголовка, а четвертый — расстояние между точками отточия. Например, при настройке `\dottedcontents{subsection}[5em]{\itshape}{3em}{1pc}` элементы уровня subsection будут печататься курсивом с отступом шириной 5em, ширина поля их номера составит 3em, а точки отточия будут следовать через 1pc.

Еще одна декларация пакета titletoc:

```
\titlecontents{уровень}[отступ]{до}{нумерованный элемент}
                {ненумерованный элемент}{заполнение}{после}
```

открывает доступ ко всем аспектам верстки элементов. Ее первый аргумент и параметр такие же, как и у предыдущей декларации. Во второй аргумент можно поместить код, который будет исполняться до вывода элемента. Два следующих аргумента предназначены для команд, настраивающих печать нумерованных и ненумерованных элементов. Последний аргумент может содержать команду, выполняющую или заменяющую отточие, а последний параметр может содержать код, который будет исполняться после вывода элемента.

Когда номера страниц в оглавлении не умещаются в отведенное под них поле, декларация `\contentsmargin[коррекция]{ширина поля}` позволяет увеличить его ширину. Ее параметр служит для выравнивания правой границы заголовков, состоящих из нескольких строк. Если последняя строка, содержащая отточие, окажется короче других строк, их длину можно уменьшить на величину, заданную параметром.

8.6. Управление плавающими объектами

Готовя к публикации статью, нет смысла тщательно подбирать место для рисунков и таблиц, их размещением займется технический

редактор. Верстая же собственные документы или книги, вывод плавающих объектов регулирует сам автор. Ранее говорилось, что параметр окружений figure и tabular позволяет указать их предпочтительное положение на странице, однако компилятор может игнорировать эти указания, если страница переполняется или оказывается недостаточно заполненной, поэтому вывод рисунков и таблиц иногда приходится настраивать, применяя специальные методы.

Автоматическим выводом плавающих объектов управляет набор параметров, представленный в табл. 8.2. В него входят счетчики, ограничивающие количество объектов на одной странице, длины, задающие отбивку рисунков и таблиц между собой и в окружающем тексте, и декларации, определяющие максимальную долю страницы, которую могут занимать рисунки и таблицы.

Настройку можно начать с этих параметров. Длины и счетчики изменяются стандартным образом, а значения долей устанавливаются с помощью команды `\renewcommand`. Например, установка счетчика bottomnumber в нуль: `\setcounter{bottomnumber}{0}`, предотвратит появления рисунков и таблиц внизу страницы и тот же эффект даст установка в нуль доли, занимаемой плавающими объектами внизу страницы: `\renewcommand\bottomfraction{0}`.

Команды `\clearpage` и `\cleardoublepage` помогают регулировать вывод плавающих объектов. Аналогично команде `\newpage` они заканчивают текущую страницу и при этом выводят все плавающие объекты, определенные до разрыва страницы, игнорируя некоторые ограничения на их размещение. При двусторонней печати команда `\cleardoublepage` следит за нумерацией страниц и при необходимости добавляет пустую, чтобы новая страница имела нечетный номер.

Разрывая страницу, указанные команды не заботятся о ее заполнении, с этой целью стандартный пакет afterpage [64] вводит команду `\afterpage{`*команды*`}`, которая заполняет страницу, а затем выполняет код, находящийся в ее аргументе. Если он содержит команду, обеспечивающую вывод плавающих объектов, они тоже выводятся, поэтому связка команд `\afterpage\clearpage` позволяет принудительно вывести рисунки и таблицы, не нарушая заполненность страниц.

[2] Параметры с именами, начинающимися префиксом «dbl», применяются при наборе в две колонки. При размере основного шрифта 12pt длины `\dblfloatsep` и `\intextsep` имеют другие величины: `\dblfloatsep` = 14pt plus 2pt minus 4pt, `\intextsep` = 14pt plus 4pt minus 4pt.

Таблица 8.2

Параметры, регулирующие вывод плавающих объектов[2]

Максимальное число плавающих объектов

Счетчики	Значения	Расположение
bottomnumber	1	внизу страницы
topnumber	2	вверху страницы
dbltopnumber	2	вверху колонки
totalnumber	3	на странице в целом

Максимальная доля страницы, занимаемая объектами

Команды	Значения	Расположение
\bottomfraction	0.3	внизу страницы
\topfraction	0.7	вверху страницы
\dbltopfraction	0.7	вверху колонки

Минимальная доля, занимаемая объектами на плавающей странице

Команды	Значения	Расположение
\floatpagefraction	0.5	на странице в целом
\dblfloatpagefraction	0.5	на странице в целом

Минимальная доля страницы, занимаемая текстом

Команда	Значение	Расположение
\textfraction	0.2	на странице в целом

Высота отбивки плавающего объекта

Длины	Значения	Разделение
\floatsep	12pt plus 2pt minus 2pt	объектов
\dblfloatsep	12pt plus 2pt minus 2pt	объектов
\textfloatsep	20pt plus 2pt minus 4pt	объектов и текста
\dbltextfloatsep	20pt plus 2pt minus 4pt	объектов и текста
\intextsep	12pt plus 2pt minus 2pt	объектов и текста

Методы, которые нужно применять для регулируемого размещения рисунка или таблицы, зависят от конкретной ситуации, но стоит отметить, что довольно часто вокруг плавающих объектов, созданных

использованием параметра `h!`, приходится делать небольшую перестановку текста, обеспечивающую нужное пространство на странице.

Когда не удается «победить» механизм автоматического размещения плавающих объектов, рисунок или таблицу можно сделать и без их участия. Как говорилось ранее, таблицу, которая, не является плавающим объектом, но может иметь подпись и получает номер в ряду других таблиц, создает окружение longtable. Отбивку такой таблицы в тексте настраивают длины `\LTpre` и `\LTpost`, задающие высоту верхнего и нижнего пробелов, по умолчанию равных упругой длине `\bigskipamount`.

С рисунком дело обстоит сложнее, его нужно полностью формировать вручную, как это сделано на примере рис. 8.1. Поместим рисунок и подпись в министраницу, чтобы они встраивались в документ как целое. В нее загрузим иллюстрацию, выровняем ее и сверстаем подпись.

Рис. 8.1. Рисунок, сверстанный вручную

Вдобавок к этому присвоим рисунку номер и поместим его подпись в список рисунков, как это делает команда `\caption` в окружении figure. Чтобы страница не была пустой, справа от рисунка выведем этот текст, помещенный в еще одну министраницу. Разберем код, реализующий все указанные действия:

```
\newcommand\figcaption[2][\lofcaption]{
   \newcommand\lofcaption{#2}
   \refstepcounter{figure}
   \addcontentsline{lof}{figure}{%
\protect\numberline{\thefigure}\ignorespaces #1}
   \small \figurename\ \thefigure. #2}
\begin{minipage}{.35\linewidth}
   \centering
   \includegraphics[width=.6\linewidth]{statement}\\[1ex]
   \figcaption{Рисунок, сверстанный вручную}
   \label{f:HandMade}
\end{minipage}
```

Компоновка рисунков и подписей уже обсуждалась в конце разд. 3.3, поэтому рассмотрим лишь определение новой команды `\figcaption`, заменяющей стандартную команду `\caption`, действующую только внутри плавающих объектов. Она имеет аргумент, печатающий подпись в тексте, и параметр, используемый для внесения ее краткого варианта в список рисунков. Аргумент и параметр команды `\figcaption`

выполняют те же функции. Значение параметра по умолчанию обеспечивает команда `\lofcaption`, в которую копируется аргумент команды `\figcaption`. Таким образом, когда параметр опущен, вместо него подставляется подпись, заданная в аргументе.

Команда `\refstepcounter` формирует номер рисунка и ссылку на него, которую команда `\label` связывает с меткой (см. разд. 7.4).

Команда `\addcontentsline` добавляет подпись в список рисунков (см. разд. 8.5). Слишком длинная строка с ее описанием разорвана. Чтобы в списке рисунков перед номером не появился лишний пробел, команда `\numberline` должна следовать непосредственно за фигурной скобкой. В команде `\figcaption` это обеспечивает символ комментария, стоящий непосредственно после скобки и сшивающий две строки, а также отсутствие пробелов в начале следующей строки перед командой `\protect`, защищающей `\numberline`. Команда `\ignorespaces` предотвращает появление лишних пробелов в подписи рисунка.

Команда `\figurename` выводит стандартное название рисунка на языке верстаемого документа (см. табл. 7.1), а команды `\thefigure` подставляют сформированный ранее номер рисунка в аргумент команды `\numberline` и подпись, печатаемую под рисунком.

Созданная команда `\figcaption` поддерживает механизм перекрестных ссылок, поэтому на рис. 8.1 можно ссылаться обычным образом.

8.7. Макет страниц

Портретная ориентацией страниц является основной. Ранее говорилось, что параметр landscape стандартных классов поворачивает страницы всего документа в альбомную ориентацию. Пакет lscape коллекции latex-graphics [21] позволяет повернуть отдельные страницы. Для этого он вводит окружение landscape . Перед изменением формата страницы выполняется команда `\clearpage`, завершающая текущую страницу и выводящая «находящиеся в очереди» плавающие объекты (см. разд. 8.6). Ориентация колонтитулов на повернутой странице не меняется.

Пакет pdflscape [65] автоматически загружает пакет lscape и обеспечивает поддержку окружения landscape в pdf-документах. Поворачивая pdf-страницу, он добавляя к ее параметрам атрибут `/Rotate`.

Подробно структура страницы документа LaTeX пока не рассматривалась. На рис. 8.2 показана ее разметка при двусторонней печати, задаваемой параметром twoside во время загрузки класса документа.

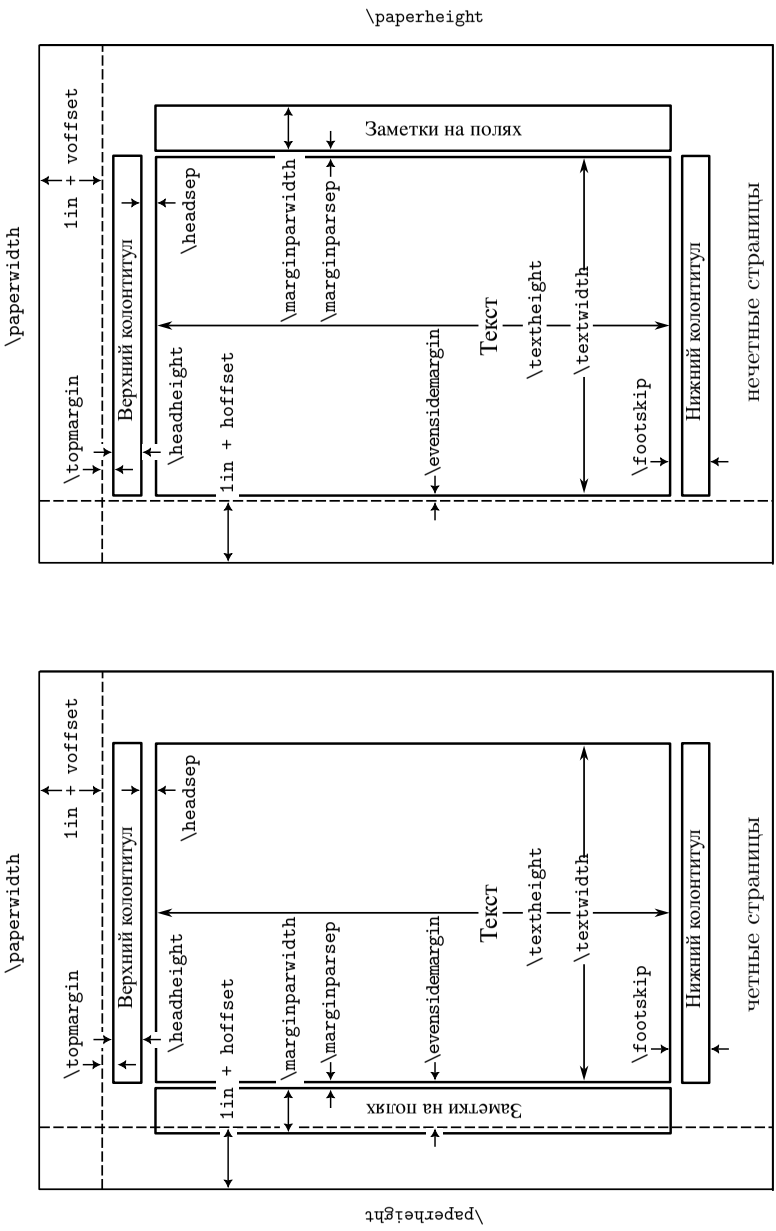


Рис. 8.2. Макет полосы набора страниц

Основными параметрами являются ширина и высота прямоугольного листа, содержащего два колонтитула, поле для заметок и прямоугольник с верстаемым текстом, который может печататься в одну или две колонки. Слева и сверху на листе отводятся пустые поля шириной в один дюйм. От их границ отсчитываются сдвиги, определяющие положение остальных элементов. В односторонней печати используется макет правых (нечетных) страниц.

Параметры страниц задают следующие длины.

`\paperwidth`, `\paperheight` — ширина и высота листа.

`\textwidth`, `\textheight` — ширина и высота текста.

`\hoffset`, `\voffset` (0pt) — длины, корректирующие ширину и высоту пустых полей, равных 1in + `\hoffset` и 1in + `\voffset`.

`\evensidemargin`, `\oddsidemargin` — расстояние от границы левого поля до левой границы текста на четных и нечетных страницах.

`\topmargin` — расстояние от границы верхнего поля до верхнего колонтитула.

`\headheight` (12pt) — высота колонтитулов.

`\headsep` — расстояние от верхнего колонтитула до верхней границы текста.

`\footskip` — расстояние от нижней границы текста до базовой линии нижнего колонтитула.

`\topskip` — расстояние от верхней границы текста до базовой линии его первой строки.

Длины `\marginparwidth` и `\marginparsep`, относящиеся к заметкам на полях, обсуждаются в следующем разделе.

При верстке в две колонки длина `\textwidth` разбивается на длины `\columnsep` (35pt) и `\columwidth = (\textwidth - \columnsep)/2`. Колонки разделяет вертикальная линия толщиной `\columnseprule`, по умолчанию равная нулю.

Изначально количество верстаемых колонок определяют параметры класса документа onecolumn и twocolumn, но с помощью деклараций `\onecolumn` и `\twocolumn[`*текст*`]` можно переключаться между верстской в одну и две колонки. Они автоматически завершают страницу командой `\clearpage` и изменяют стиль верстки, при этом *текст* параметра `\twocolumn` печатается в начале новой страницы в одну колонку. Отметим, что данные декларации являются хрупкими.

Параметр декларации \twocolumn дает возможность совместить на одной странице набор с разным количеством колонок, однако с его помощью нельзя начать ее двумя колонками, а закончить одной. Это ограничение снимает стандартный пакет multicol. Он вводит новое окружение multicols:

\begin{multicols}{число колонок}[текст][высота],

позволяющее верстать до десяти колонок текста. При его простейшем использовании достаточно указать число колонок, ширина которых будет автоматически вычислена из длины строки предшествующего текста. Текст, верстаемый окружением multicols, отбивается от окружающего текста полями высотой 12 pt. В первом параметре окружения можно указать текст, который аналогично декларации \twocolumn будет напечатан в одну колонку. Если верстаемый текст не умещается на странице, он переносится на следующую,[3] при этом по умолчанию его высота на первой странице не должна быть меньше шести длин \baselineskip. Второй параметр окружения позволяет отрегулировать это значение.

В окружениях multicols можно размещать сноски, вынесенные формулы и даже плавающие объекты figure* и table*, занимающие ширину страницы. Верстку multicols настраивают большое количество параметров и команд, которые заинтересованный читатель найдет в документации пакета multicol [66]. Часть из них описана в книгах [3, 4].

Отметим, что при многоколоночном наборе, загрузив пакет ftnright [67], все сноски \footnote{•} можно вывести в правую колонку.

8.8. Заметки на полях

Заметки на полях редко используются в бумажной печати, так как ширина пустых полей на страницах обычно невелика. Однако в электронных документах, где она ограничена только шириной экрана мониторов, в заметках можно разместить сноски и даже небольшие рисунки.

При двусторонней печати заметки четных страниц выносятся на левое поле, а нечетных страниц — на правое, как показано на рис. 8.2. Для

[3] Распределяя текст по страницам при двухколоночном наборе, LaTeX рассматривает колонки как страницы, однако окружение multicols так не поступает. Перенося текст на следующую страницу, оно помещает его в ту же колонку, а соседнюю оставляет пустой.

этого ширина обоих полей должна быть одинаково большой. Увеличить их можно следующим образом.[4]

Обнулим длины `\oddsidemargin` и `\hoffset`, тогда ширина правого поля нечетных страниц будет равна ширине листа за вычетом одного дюйма и ширины текста `\textwidth`. Установим длину `\evensidemargin`, равной этому значению за вычетом одного дюйма, и на четных страницах получим левое поле той же ширины. Теперь можно установить ширину поля записей `\marginparwidth`. Ее не стоит делать более трети или четверти ширины текста, так как широкие записи смотрятся как дополнительная колонка текста, а слишком узкие плохо выравниваются.

Ширину поля заметок можно еще более увеличить, уменьшив поле переплета страниц. Для этого сдвинем влево линию отсчета полей, показанную пунктирной вертикальной линией на рис. 8.2, задав отрицательное значение длины `\hoffset`, например

`\setlength\hoffset{-0.5in}`.

Затем добавим к `\evensidemargin` ее удвоенное положительное значение. Если после этих операций места для заметок не хватает, или наоборот пустые поля слишком велики, в электронном документе можно скорректировать ширину листа `\paperwidth`.

При односторонней печати все заметки выводятся на правое поле страницы. При верстке в две колонки заметки к тексту левых колонок выводятся слева, а для правых — справа. Однако в документе, верстаемом в несколько колонок, вряд ли стоит делать заметки. Декларация `\reversemarginpar` позволяет инвертировать вывод заметок. После ее выполнения заметки, выводившиеся слева, будут печататься справа и наоборот. Декларация `\normalmarginpar` возвращает нормальный порядок вывода заметок.

Примеры заметок на полях и полезные замечания, касающиеся их верстки, приведены на рис. 8.4. Чтобы отвести поле для заметок не в ущерб основному тексту, содержащая их страница сделана альбомной.

Заметку на полях создает команда

`\marginpar[`*текст левого поля*`]{`*текст правого поля*`}`.

Ее аргумент содержит текст заметки. Текст параметра выводится в заметку, попавшую на левое поле, а если он опущен, как всегда, используется аргумент. Это позволяет по-разному настроить параметры

[4] Напомним, что удобный интерфейс установки всех параметров страницы дает пакет `geometry`.

Заметка, верстаемая по умолчанию, выглядит как дополнительная узкая колонка текста.

Заметки смотрятся хорошо, если не очень длинный текст набран мелким шрифтом и выровнен по внешней стороне страницы.

Рис. 8.3. Пасутся тигры на полях

Ширина поля, отводимого для заметок, невелика, поэтому имеет смысл уменьшить размер их шрифта. Но даже в таком виде текст, по умолчанию выравниваемый с двух сторон, может смотреться не очень хорошо из-за большого количества переносов слов. Заметка, состоящая из целых слов и выровненная по внешней стороне страницы, выглядит много лучше. Такое форматирование обеспечивают декларации `\raggedleft` и `\raggedright`. Исходя из сказанного, для верстки заметок удобно ввести новую команду:

```
\newcommand}\marginote[1]{
   \marginpar[\footnotesize}\raggedright #1]{\footnotesize\raggedleft #1}},
```

которая на левом поле выравнивает текст слева, а на правом — справа.

Для сравнения верхняя заметка сверстана стандартной командой `\marginpar`, а следующая — командой `\marginote`.

В виде заметок можно оформить даже рисунок, который придется встраивать вручную, как описано в конце предыдущего раздела, так как окружение figure в заметку вставить не удастся.

Аргумент команды `\marginpar` в приведенном на полях рис. 8.3 содержит иллюстрацию и подпись, оформленную с помощью введенной ранее команды `\figcaption`, автоматически добавившую иллюстрацию в список рисунков. Такой трюк делает документ нестандартным, но рукописи книг обычно содержат много нестандартных решений, поэтому использование здесь «авторской» команды вполне оправданно.

В настоящее время многие зарубежные издательства эффективно используют заметки на полях электронных книг, размещая в них сноски, небольшие иллюстрации и подписи к рисункам.

Рис. 8.4. Примеры заметок на полях

верстки заметок на четных и нечетных страницах. Зазоры шириной `\marginparsep` отделяют заметки от основного текста.

Компилятор стремится разместить заметку на уровне текста, который она поясняет. Если команда `\marginpar` стоит внутри абзаца, первая строка заметки выводится на уровне содержащей ее строки. Когда она находится между абзацами, заметка размещается на уровне последней строки предшествующего абзаца. Если же `\marginpar` окажется внизу страницы, компилятор поднимет заметку, стремясь оставить ее на текущей странице. Такие правила действуют, когда заметок немного и они не накладываются друг на друга. В противном случае соседние заметки разделяются зазором высотой `\marginparpush`, более нижние смещаются вниз и даже могут оказаться на другой странице.

8.9. Настройка колонтитулов

Простейшие колонтитулы используются для нумерации страниц, в книгах в них часто приводят названия глав и разделов, а в журналах список авторов и заголовки статей. Содержание колонтитулов регулирует декларация `\pagestyle{`*стиль страницы*`}`. Обычно ее используют в преамбуле, чтобы установить стиль для всего документа. Команда `\thispagestyle{`*стиль страницы*`}` позволяет изменить колонтитулы той страницы, на которой она находится. *Стиль страницы* имеет следующие значения:

`empty` — колонтитулы отсутствуют;

`headings` — нижний колонтитул пуст, а верхний содержит номер страницы и название раздела;

`myheadings` — нижний колонтитул пуст, а верхний содержит номер страницы и текст, задаваемый автором, который по умолчанию отсутствует;

`plain` — верхний колонтитул пуст, а нижний содержит номер страницы.

По умолчанию класс `book` использует стиль `headings`, а остальные классы — стиль `plain`.

Формирование колонтитулов — сложный процесс, состоящий из нескольких этапов, которые мы рассмотрим сначала «снизу вверх», чтобы описать участвующий в нем набор команд, а потом дадим анализ самого процесса «сверху вниз», чтобы понять, как им управлять.

Текст колонтитулов выводят команды

```
\@evenfoot{текст},   \@oddfoot{текст},
\@evenhead{текст},   \@oddhead{текст}.
```

Две первые печатают нижний колонтитул, две последние — верхний.

В стиле plain команды `\@evenfoot` и `\@oddfoot` содержат счетчик страниц, зажатый пружинами:

```
\def\@oddfoot{\hfil\thepage\hfil},
\let\@evenfoot\@oddfoot.
```

Как следствие, номера выводятся внизу по центру страниц. В остальных стилях команды `\@evenfoot` и `\@oddfoot` пусты, поэтому нижние колонтитулы отсутствуют.

В стилях plain и empty пусты команды `\@evenhead` и `\@oddhead`. В стилях headings и myheadings они определены как:

```
\def\@evenhead{\thepage\hfil\slshape\leftmark}.
\def\@oddhead{{\slshape\rightmark}\hfil\thepage}.
```

Поэтому на четных страницах слева выводится номер, а справа — текст, находящийся в команде `\leftmark`, печатаемый наклонным шрифтом. Такое выравнивание обеспечивает пружина, стоящая между номером и текстом. На нечетных страницах номер и текст, находящийся теперь в команде `\rightmark`, меняются местами.

Команды `\leftmark` и `\rightmark` формирую декларации

```
\markboth{левый колонтитул}{правый колонтитул},
\markright{колонтитул}.
```

Команда `\markright` копирует свой аргумент в `\rightmark`. Первый и второй аргументы команды `\markboth` заносятся, соответственно, в `\leftmark` и `\rightmark`.

Рассмотрим формирование колонтитулов в стиле headings при односторонней печати, которая подразумевает одинаковое оформление страниц, обеспечиваемое декларацией `\markright`.

В классах book и report, создавая главу, компилятор выполняет команду `\chaptermark{краткий заголовок}`, аргументом которой является краткий заголовок главы, указанный в параметре команды `\chapter`, или ее полный заголовок, если параметр отсутствует. При односторонней печати эта команда определена как:[5]

[5] Определения воспроизводятся не буквально, часть системных команд заменена знакомыми читателю аналогами, описанными ранее.

```
\def\chaptermark #1 {\markrigh
            {\MakeUppercase{\chaptername\ \thechapter. #1}}},
```

Декларация `\markrigh` заносит в команду `\rightmark` текст, печатаемый прописными буквами, состоящий из стандартного заголовка `\chaptername`, номера главы `\thechapter` и ее заголовка, находящегося в аргументе `#1` и отделенного от номера точкой. В классе article ту же функцию при создании раздела выполняет команда

`\sectionmark{`*краткий заголовок*`}` :

```
\def\sectionmark #1 {\markrigh{\MakeUppercase{\thesection\quad #1}}}.
```

Она помещает в `\rightmark` номер и заголовок раздела, печатаемый прописными буквами. Таким образом, текст колонтитулов обновляется для каждой новой главы, или раздела.

Ранее не встречавшиеся команды `\MakeUppercase{`*текст*`}` и `\MakeLowercase{`*текст*`}` переводят строчные буквы в прописные и наоборот.

При двусторонней печати четные и нечетные страницы оформляются по-разному, поэтому определение команды `\chaptermark` меняется следующим образом:

```
\def\chaptermark #1 {\markboth
            {\MakeUppercase{\chaptername\ \thechapter. #1}}{}},
```

Теперь она задает текст левого колонтитула и «очищает» правый. Его текст при создании разделов формирует команда `\sectionmark`, определение которой претерпевает лишь слабое изменение, заключающееся в замене пробела `\quad` точкой:

```
\def\sectionmark #1 {\markrigh {\MakeUppercase{\thesection. #1}}}.
```

В результате выполнения обеих команд в левый колонтитул помещаются названия и номера глав, а в правый — названия и номера разделов.

Класс article заменяет определение команды `\sectionmark` на:

```
\def\sectionmark #1 {\markboth
            {\MakeUppercase{\thesection\quad #1}}{}},
```

и вводит команду `\subsectionmark{`*краткий заголовок*`}` , выполняемую при создании раздела `\subsection`:

```
\def\subsectionmark #1 {\markrigh
            {\MakeUppercase{\thesubsection\quad #1}}}.
```

В результате их действия в левый колонтитул помещаются названия и номера разделов, а в правый — названия и номера подразделов.

Разберем теперь, какую последовательность действий влечет изменение стиля страниц. Прежде всего отметим, что стиль, заданный в преамбуле, можно корректировать «по месту» в тексте, но, как правило, во всем документе выдерживается единый стиль.

8.9.1. Стиль empty

Декларация `\pagestyle{empty}` очищает аргументы команд, печатающих колонтитулы, `\@evenfoot`, `\@oddfoot`, `\@evenhead` и `\@oddhead`, и более они автоматически не обновляются. Назовем такое действие *аннулированием*. Если после этого отредактировать команды вручную, внесенный в них текст появится в колонтитулах. Например, после установки `\def\@evenhead{\itshape\hfil-- Четные страницы --\hfil}` вверху по центру четных страниц будет выводиться надпись

– Четные страницы –.

Здесь и далее, чтобы не загромождать примеры, мы опускаем команды, меняющие статус символа `@`.

8.9.2. Стиль plain

Декларация `\pagestyle{plain}` аннулирует аргументы команд `\@evenhead` и `\@oddhead`, а команды `\@evenfoot` и `\@oddfoot` настраивает на печать номеров по центру внизу страниц, как это описано ранее. Формат печати задает декларация `\pagenumbering{`*формат*`}`. *Форматами* служат имена команд, перечисленных в табл. 7.2. Стандартом являются арабские цифры, но иногда для вводной части книг используются большие и малые римские цифры.

Декларация `\pagenumbering` начинает новый отсчет страниц, устанавливая значение их счетчика в единицу и меняя формат его вывода.

Если в преамбуле документа задать `\pagenumbering{roman}`, а перед первой главой — `\pagenumbering{arabic}`, то страницы вводной части будут нумероваться малыми римскими цифрами, а нумерация первой главы начнется с обычной единицы.

Редактируя `\@evenfoot` и `\@oddfoot`, можно настроить остальные аспекты вывода номеров, например напечатать их мелким шрифтом в левом углу на четных страницах и в правом на нечетных, как это делают определения:

```
\def\@evenfoot{\small \thepage \hfil},
\def\@oddfoot{\small \hfil \thepage}.
```

Команда `\pagenumbering{gobble}` полностью подавляет нумерацию страниц. Номера отсутствуют не только в колонтитулах, но и оглавлении, ссылках и т. д. Пакет hyperref восстанавливает нумерацию, отмененную таким образом. Работая с ним, используйте с этой целью стиль `\pagestyle{empty}`.

8.9.3. Стиль headings

Декларация `\pagestyle{headings}` аннулирует аргументы команд `\@evenfoot` и `\@oddfoot`, а в аргументы команд `\@evenhead` и `\@oddhead` помещает команды `\leftmark` и `\rightmark` вместе с номерами страниц в формате, прежде заданном `\pagenumbering`.

Описанный ранее механизм формирования текста команд `\leftmark` и `\rightmark` обеспечивает следующее оформление. Первая страница главы содержит только номер, а на следующих четных страницах в колонтитулах печатается также название и номер главы. До создания первого раздела колонтитулы нечетных страниц содержат только их номера, а потом к ним добавляется номер и название раздела.

Разберем настройку колонтитулов на примере их оформления в данной книге. В левом углу четных страниц выводится номер станицы, по центру маленьким шрифтом прописными буквами — название главы, а в правом углу — ее номер римскими цифрами. На нечетных страницах номер страницы выводится справа, по центру печатается название раздела, а слева его номер. Сами же колонтитулы отделяются от текста сплошной линией.

Отчеркивание затрагивает команды печати колонтитулов, поэтому определим, прежде всего, их:

```
\def\@evenhead{\underline{
    \makebox[\textwidth]{\thepage\hfill\leftmark}}}
\def\@oddhead{\underline{
    \makebox[\textwidth]{\rightmark\hfill\thepage}}}
```

Для отчеркивания применена команда `\underline{•}`, имеющая существенную в данном случае особенность: в подчеркнутом тексте не действуют пружины, необходимые для выравнивания колонтитула. Поэтому аргументом `\underline` служит команда `\makebox`, создающая бокс с шириной, равной ширине страницы. В бокс вставлены номер страницы и текст колонтитула, разделенные жесткой пружиной.

Теперь отредактируем команды, формирующие текст колонтитулов:

```
\def\chaptermark #1{\markboth{
    \small\MakeUppercase{#1} \hfill ГЛ.\,\Roman{chapter}}{}}
\def\sectionmark #1{\markright{§\,\thesection \hfill #1}}
```

В команду `\leftmark` декларация `\markboth` помещает команды `\small` и `\MakeUppercase`, обеспечивающие нужное оформление названия главы `#1`, жесткую пружину `\hfill`, отделяющую его от заголовка `ГЛ.` и номера главы, печатаемого командой `\Roman`. Заголовок и номер разделены тонким пробелом `\,`.

Текст правого колонтитула, который декларация `\markright` помещает в команду `\rightmark`, начинается символом §, тонким пробелом и номером раздела `\thesection`, за которыми следуют жесткая пружина и название раздела `#1`.

В результате «сборки» колонтитулов в командах `\@evenhead` и `\@oddhead` названия глав и разделов центрируются пружинами, отжимающими номера страниц, глав и разделов на края страниц. Жесткие пружины применяются, чтобы нивелировать действие мягких пружин `\hfil`, которые неявно используются, если допустить автоматическое выравнивание текста в боксе `\makebox`.

8.9.4. Стиль myheadings

Декларация `\pagestyle{myheadings}` аннулирует аргументы команд `\@evenfoot` и `\@oddfoot`, а в аргументы команд `\@evenhead` и `\@oddhead` помещает команды `\leftmark` и `\rightmark` вместе с номерами страниц в формате, прежде заданном `\pagenumbering`. При этом текст из команд `\leftmark` и `\rightmark` убирается, а его автоматическое обновление при создании новых разделов отключается. В итоге автоматически формируемые колонтитулы содержат лишь номера страниц, разнесенные по левым и правым углам на четных и нечетных страницах, но авторам дана возможность вручную задать текст команд `\leftmark` и `\rightmark` с помощью деклараций `\markboth` и `\markright`.

Стиль myheadings позволяет настроить колонтитулы разделов, формируемых командами `\chapter*`, `\section*` и `\subsection*`. При их создании текст команд `\leftmark` и `\rightmark` аннулируется, так как по умолчанию их название в колонтитулы не вносится, и при необходимости это автор должен сделать сам.

Приведем пример. Согласно правилам российской полиграфии введение и заключение книги не нумеруются, поэтому их удобно оформлять, используя раздел `\section*`. Тогда, установив стиль myheadings, после команды `\section*` нужно поставить команду `\markboth{•}{•}` с названием раздела •, которое попадет в колонтитулы[6].

Не забудьте вновь включить стиль headings непосредственно *перед* созданием следующего нумерованного раздела.

Настройка колонтитулов — специфическая задача, возникающая довольно редко. Описанные средства позволяют с ней справиться, но если их окажется недостаточно, воспользуйтесь пакетом fancyhdr [68], позволяющим верстать колонтитулы любой сложности.

8.10. Настройка списков

В первой части книги подробно разобрана верстка библиографии, описано создание нумерованных и ненумерованных списков. За кадром остались вопросы, связанные с настройкой формата вывода списков, разбираемые далее.

Описанные в разд. 5.1 списки itemize, enumerate и description имеют общие настройки, так как они суть модификации одного и того же окружения list, служащего основой еще целого ряда окружений. Поэтому сначала мы разберем его устройство, а потом посмотрим, как оно используется для создания списков и других конструкций.

Окружение list представляет собой список с довольно сложной внутренней структурой, представленной на рис. 8.5, настраиваемой большим числом параметров:

`\begin{`*list*`}{`*метка*`}{`*настройки*`}`.

По умолчанию записи списка используют *метку*, заданную в первом аргументе окружения. Во втором аргументе можно настроить команду, выводящую метку, и установить параметры, управляющие версткой *списка*. Параметры представляют собой длины, задающие горизонтальные и вертикальные отбивки элементов списка, показанных на рис. 8.5. Значения длин приведены в табл. 8.3 и 8.4.

Предваряя анализ настроек, отметим, что в вертикальной отбивке участвуют упругие длины, поэтому высота отбивок может меняться в довольно широких пределах.

[6] В книге [2] ее рекомендуется ставить без пробела после первого слова раздела.

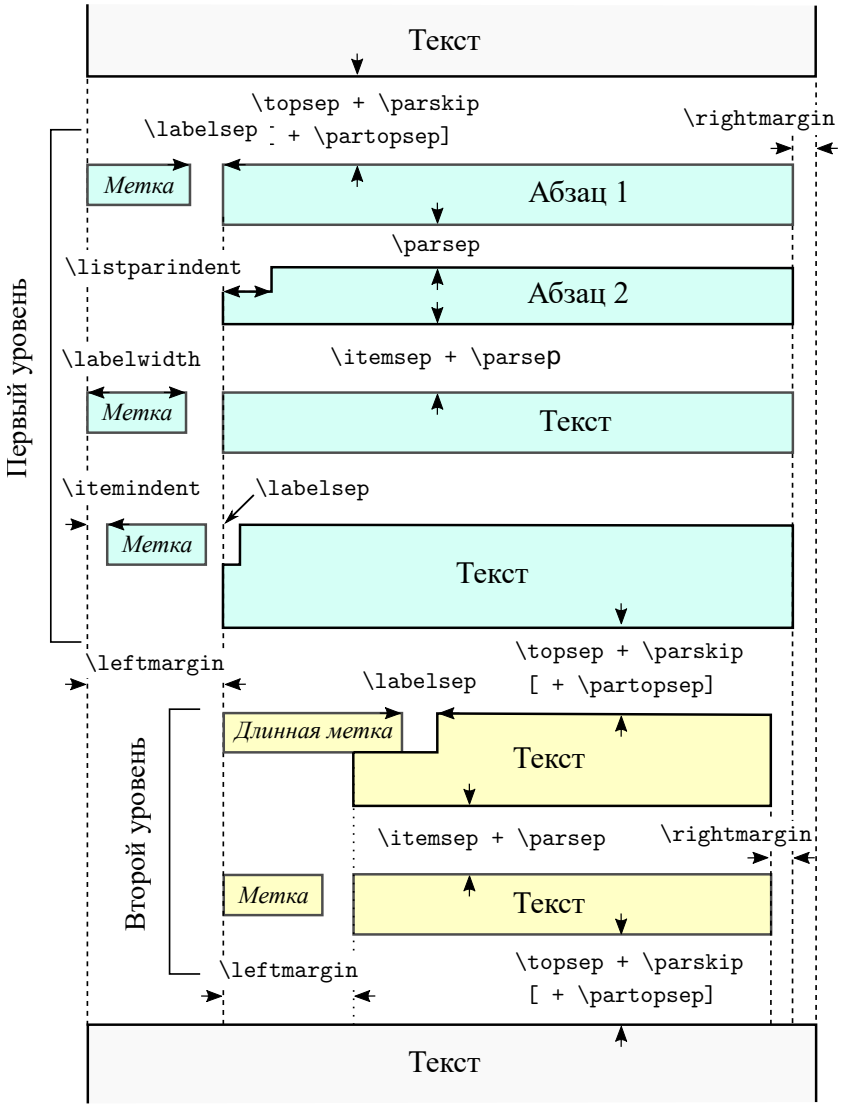


Рис. 8.5. Формат стандартных списков

В разд. 5.1 говорилось, что списки itemize и enumerate имеют четыре уровня вложенности, а у description — их шесть. Величины длин

варьируются в зависимости от уровня вложенности, размера шрифта, которым набирается список, и размера основного шрифта документа. Различные комбинации этих параметров представлены в табл. 8.3.

Декларации `\@listi`, `\@listii` и `\@listiii` задают длины для трех уровней списков, а начиная с четвертого, используются настройки третьего уровня. Приведем для примера определение одной из них:

```
\def\@listii {\leftmargin \leftmarginii
              \labelwidth \leftmarginii
              \advance\labelwidth -\labelsep
              \topsep  4.5pt plus 2pt minus 1pt
              \parsep  2pt   plus 1pt minus 1pt
              \itemsep \parsep}
```

В нем используется синтаксис присваивания значений TEX:

`команда =` *значение*,

`\advance команда by` *значение*.

Команда может быть длиной или счетчиком, *значение* можно задать явно или другой командой. Очень часто знак равенства и ключевое слово `by` опускают. Например, в приведенном определении значение длины `\leftmarginii` присваивается длинам `\leftmargin` и `\labelwidth`, а затем последняя уменьшается на длину `\labelsep`.

В табл. 8.3 собраны длины, регулирующие отбивки элементов списков, как показано на рис. 8.5. Их значения указаны для кеглей `10pt`, `11pt` и `12pt`, а также для шрифтов уменьшенных декларациями `\small` и `\footnotesize`. Команды, стоящие на месте значений, указывают на эквивалентность длин. Например, длине `\itemsep` присваивается значение `\parsep` или `\topsep`, а длина `\partopsep` при уменьшении шрифта не меняется.

Запись списка может содержать несколько абзацев, разделяемых зазором высотой `\parsep`. Чтобы записи хорошо выделялись зрительно, их отбивка увеличивается на высоту `\itemsep`. Еще более широкий зазор отделяет список от окружающего текста. Его высота складывается из длин `\topsep` и `\parskip`. Последняя добавляется к межстрочному интервалу между абзацами текста как внутри, так и вне списков. Ее величина представлена в нижней части таблицы вместе с горизонтальными длинами. Если список стоит перед абзацем или следует за ним, т. е. перед ним или после него находится пустая строка или заменяющая ее команда `\par`, отбивка увеличивается на высоту `\partopsep`.

Таблица 8.3

Параметры настройки стандартных списков: длины вертикальных отбивок

Кегль	Длины	\@listi	\@listii	\@listiii	\small	\footnotesize
12pt	\topsep	10pt +4pt −6pt	5pt +2.5pt −1pt	2.5pt ±1pt	9pt +3pt −5pt	6pt ±2pt
	\parsep	5pt +2.5pt −1pt	2.5pt ±1pt	0pt ±0pt	4.5pt +2pt −1pt	3pt +2pt −1pt
	\partopsep	3pt ±2pt	3pt ±2pt	1pt +0pt −1pt	\partopsep	\partopsep
	\itemsep	5pt +2.5pt −1pt	\parsep	\topsep	\parsep	\parsep
11pt	\topsep	9pt +3pt −5pt	4.5pt +2pt −1pt	2pt ±1pt	6pt ±2pt	4pt ±2pt
	\parsep	4.5pt +2pt −1pt	2pt ±1pt	0pt ±0pt	3pt +2pt −1pt	2pt ±1pt
	\partopsep	3pt ±1pt	3pt ±1pt	1pt ±0pt	\partopsep	\partopsep
	\itemsep	4.5pt +2pt −1pt	\parsep	\topsep	\parsep	\parsep
10pt	\topsep	8pt +2pt −4pt	4pt +2pt −1pt	2pt ±1pt	4pt ±2pt	3pt ±1pt
	\parsep	4pt +2pt −1pt	2pt ±1pt	0pt ±0pt	2pt ±1pt	2pt ±1pt
	\partopsep	2pt ±1pt	2pt ±1pt	1pt +0pt −1pt	\partopsep	\partopsep
	\itemsep	4pt +2pt −1pt	\parsep	\topsep	\parsep	\parsep

Таблица 8.4

Параметры настройки стандартных списков: длины горизонтальных отбивок

Общие				Одна колонка		Две колонки	
Команда	Длина	Команда	Длина	Команда	Длина	Команда	Длина
\labelsep	0.5em	\leftmarginii	2.2em	\leftmargini	2.5em	\leftmargini	2em
\bibindent	0.5em	\leftmarginiii	1.87em	\leftmarginv	1em	\leftmarginv	0.5em
\parskip[7]	0pt +1pt	\leftmarginiv	1.7em	\leftmarginvi	1em	\leftmarginvi	0.5em

[7] Вертикальный размер.

Версткой строк управляют неупругие длины, величины которых выражаются в единицах em, привязанных к текущему размеру шрифта. Первая строка записи, содержащая метку, верстается по собственным правилам, а остальные — как обычный текст. Длина строк ограничена слева и справа пустыми полями, шириной `\leftmargin` и `\rightmargin`. Как показано на рис. 8.5, поля списков первого уровня отсчитываются от краев страницы, или границ предшествующего абзаца, а поля вложенных списков — от границ предыдущего списка. По умолчанию `\rightmargin` устанавливается в нуль. Случаи, в которых это значение изменяется, мы оговорим особо. Во вложенных списках длина `\leftmargin` получает значение `\leftmargin`•, где • ≡ i–vi обозначает римскую цифру, соответствующую уровню вложенности. Значения длин `\leftmargin`• приведены в табл. 8.4.

Метка записи выносится на левое пустое поле и помещается в бокс шириной `\labelwidth`, равной длине `\leftmargin` за вычетом длины `\labelsep`, отделяющей метку от текста записи. Если метка оказывается длиннее пустого поля, ширина бокса увеличивается так, чтобы она полностью в него уместилась. В этом случае метка, вклинившись в текст записи, отделяется от него зазором `\labelsep`, как показано внизу рис. 8.5. Строка с меткой имеет отступ шириной `\itemindent`, величина которого добавляется к длинам `\labelwidth` и `\labelsep`. Если он не равен нулю, текст первой строки отступает от границы пустого поля на ту же длину. Запись может содержать несколько абзацев текста. Их отступ регулирует длина `\listparindent`. По умолчанию значения `\itemindent` и `\listparindent` устанавливаются в нуль.

На входе в окружение list вычисляется уровень вложенности списка, и если он не превышает шести, выполняется декларация

`\@list`•`{`*установка длин*`}`, где • ≡ i–vi,

устанавливающая величины перечисленных ранее длин. Их значения задает класс документа. Уровень вложенности хранит счетчик `\@listdepth`.

Метку формирует и выводит команда `\makelabel{`*метка*`}`, определенная в окружении list. Ее аргументом являются текст или символ метки и команды форматирования, настраиваемые отдельно для каждого типа списка. В качестве метки может выступать значение счетчика, например номер в списке enumerate, тогда декларация `\usecounter{`*счетчик*`}` привязывает к списку *счетчик*. Она должна присутствовать во втором аргументе окружения list, а счетчик должен быть определен до входа в окружение. Декларация `\usecounter`

устанавливает счетчик в нуль, а каждая команда \item обновляет его значение с помощью команды \refstepcounter, обеспечивающей возможность его использования в системе перекрестных ссылок (см. разд. 1.5.6).

Разберем теперь определения окружений LaTeX, созданных на основе списка list, но прежде скажем, как менять их настройки *после* определения. Если между командами \begin{имя окружения} и \item, т. е. перед первой записью, отредактировать значения указанных выше длин, формат вывода списка изменится, при этом сделанные изменения не затронут остальные списки. Чтобы настроить верстку всех списков, декларации \@list• нужно изменить в преамбуле.

Представленные далее определения содержат компактные нотации TeX, используемые в оригинальных кодах. Структура окружений воспроизводится корректно, но не буквально. Часть команд, несущественных для понимания конструкции в целом, не приводится. Любознательный читатель найдет их в документации ядра LaTeX [48] или коде стандартных классов.

8.10.1. Окружение itemize

Окружение itemize помечает записи метками, которые печатают команды \labelitemi , \labelitemii , \labelitemiii , \labelitemiv . Все они устроены одинаково:

\labelitem• ≡ {\labelitemfont символ}.

Символы устанавливает класс документа, подобно тому, как это делает декларация \itemlabelset на с. 164. В стандартных классах метками служат символы • (i), – (ii), ∗ (iii), · (iv), а шрифт \labelitemfont соответствует шрифту \normalfont.

Окружение itemize устроено следующим образом:

```
\newenvironment{itemize}{ Контроль уровня вложенности
                \begin{list}{\labelitem•}{
                    \def\makelabel} ##1{\hss \llap{##1}} }
              }{\end}{list}}.
```

Сначала вычисляется уровень вложенности окружения •, и если он не превышает четырех, формируется окружение list с меткой \labelitem•. Команда \makelabel во втором аргументе окружения list определена так, чтобы метки, помещаемые в бокс \llap, прижимались справа к пробелу \labelsep. Это делают пружина \hss и бокс \llap (см. разд. 7.3.1). Обозначение ##1 передает компилятору обра-

щение к аргументу команды, а не к аргументу окружения. На выходе из окружения itemize выполняется команда \end{list}.

Таким образом, окружение itemize, по сути, является окружением list с описанными выше настройками.

Отметим, что пружины \hfil и \hfill, помещенные в параметр команды \item, не могут регулировать выравнивание меток. Чтобы получить такую возможность, нужно переписать команду \makelabel. Например, определение

```
\renewcommand\makelabel[1]{\hfil\bfseries #1},
```

сохранив правое выравнивание, сделает метку жирной и позволит команде \item[*символ* \hfil] сместить ее в центр бокса или, при использовании пружины \hfill, к его левой границе.

8.10.2. Окружение enumerate

Метки нумерованных списков настраивает класс документа, делая это в два этапа. Сначала определяются команды вывода счетчиков:

```
\renewcommand \theenumi{\arabic{enumi}},
\renewcommand \theenumii{\alph{enumii}},
\renewcommand \theenumiii{\roman{enumii}},
\renewcommand \theenumiv{\Alph{enumiv}},
```

а затем они используются для задания меток:

```
\newcommand \labelenumi{\theenumi.},
\newcommand \labelenumii{(\theenumii)},
\newcommand \labelenumiii{\theenumii.},
\newcommand \labelenumiv{\theenumiv.}.
```

Кроме этого определяются префиксы, выводимые перед номерами записей вложенного списка, в ссылках \ref:

```
\renewcommand \p@enumii{\theenumi},
\renewcommand \p@enumiii{\theenumi(\theenumii)},
\renewcommand \p@enumiv{\p@enumiii\theenumiii}.
```

В соответствии с ними и определениями команд \theenum• ссылка на запись списка четвертого уровня будет состоять из номера записи первого уровня, указанного арабскими цифрами, стоящей за ним в круглых скобках строчной буквы, соответствующей номеру записи второго уровня, за которой следует номер записи третьего уровня, набранный малыми римскими цифрами, и, наконец, прописная буква, соответствующая номеру самой записи, например 2(a)iC.

Список enumerate определен аналогично списку itemize:

```
\newenvironment{enumerate}{ Контроль уровня вложенности
        \begin{list}{\labelenum•}{
          \usecounter{enum•} \def\makelabel ##1{\hss \llap{##1}} }
    }{\end{list}},
```

с той лишь разницей, что метки задают команды `\labelenum•`, а второй аргумент окружения list содержит дополнительно команду `\usecounter`, связывающую список со счетчиком enum•. Таким образом, все сказанное о настройках списка itemize напрямую относится и к enumerate.

Пакет enumerate [69] коллекции tools позволяет настроить вывод номеров списка enumerate. Он добавляет окружению параметр:

`\begin{enumerate}[`*текст и формат*`]`.

В нем можно указать *текст*, который нужно вывести перед записями, и *формат* печати номера. Например записи окружения, заданного командой `\begin{enumerate}[№1.]`, будут нумероваться как «№1.», «№2.» и т. д. Буквы A (`\Alph`), a (`\alph`), I (`\Roman`), i (`\roman`) и цифра 1 (`\arabic`) задают форматы печати номеров, указанные в скобках. Если же они содержатся в *тексте*, их нужно заключить в фигурные скобки, как в данном примере: `\begin{enumerate}[{example} I)]`. Отметим, что ссылки, генерируемые командами `\ref`, будут выводить только номера без *текста*.

8.10.3. Окружение description

Окружения itemize и enumerate вводятся в ядре LaTeX, а список description определяет класс документа. В стандартных классах вложенность окружения description контролирует окружение list, которое ограничивает ее шестым уровнем.

Окружение description предназначено для создания списков терминов, служащих метками записей. У каждой записи термин свой, поэтому параметры меток изменяются от записи к записи и для их печати вводится команда `\descriptionlabel{`*термин*`}`. Разберем ее определение и устройство списка description:

```
\newcommand*\descriptionlabel[1]{
                          \hspace\labelsep \normalfont\bfseries #1}
\newenvironment{description}{
                          \begin{list}{}{
                          \labelwidth 0pt \itemindent -\leftmargin
```

```
                \let \makelabel \descriptionlabel} }
                }{\end{list}}.
```

В определении description первый аргумент окружения list, в который помещается метка, пуст, а во втором аргументе ширина бокса с меткой устанавливается в нуль. Отрицательная длина отступа `\itemindent` смещает метку влево на ширину пустого поля `\leftmargin`. Присваивание `\let...` делает команду ее печати `\makelabel` синонимом команды `\descriptionlabel`. Последняя устанавливает для вывода метки жирный шрифт и вставляет перед ней пробел шириной `\labelsep`. Как следствие, ширина бокса с меткой, первоначально равная нулю, становится равной сумме длины `\labelsep` и длины термина. Так как пробелы, находящиеся перед термином, игнорируются, а ширина бокса фиксирована, при печати термин смещается в начало бокса, а пробел становится за ним.

В результате действия всех команд термин, смещенный на границу пустого поля, печатается жирным шрифтом, а пробел `\labelsep` отделяет его от последующего текста.

8.10.4. Окружение thebibliography

Список литературы также формирует окружение list, использующее вместо команды `\item` команду `\bibitem`. Окружение thebibliography определяет класс документа. Разберем его устройство:

```
\newenvironment{thebibliography}[1]{
          \section*{\refname}
          \markboth{\MakeUppercase\refname}{\MakeUppercase\refname}
          \begin{list}{ \@biblabel{\arabic{enumiv}} }{
             \settowidth\labelwidth{\@biblabel{#1}}
             \leftmargin\labelwidth \advance\leftmargin \labelsep
             \usecounter{enumiv}
             \renewcommand\p@enumiv{}
             \renewcommand\theenumiv{\arabic{enumiv}}
             \@openbib@code }
          }{\end{list}}.
```

Сначала команда `\section*` печатает заголовок библиографии,[8] а команда `\markboth` заносит его в колонтитулы, переведя буквы в заглавные. Затем формируется окружение list. В качестве метки ему

[8] Здесь приведено определение класса article.cls. В классах book.cls и report.cls команда `\section*{\refname}` заменяется на `\chapter*{\bibname}`.

передается команда \@biblabel, печатающая номер источника арабскими цифрами. Счетчик enumiv используется для нумерации списка литературы.

Во втором аргументе окружения list задаются параметры верстки источников. Ширина бокса с меткой \labelwidth устанавливается равной ширине текста, находящегося в аргументе #1 окружения thebibliography, печатаемого командой \@biblabel (см. разд. 5.2.1 и с. 159). Ширина левого поля \leftmargin устанавливается равной сумме длин \labelwidth и \labelsep. Счетчик enumiv привязывается к списку литературы, а формат его вывода корректируется. Для этого его префикс \p@enumiv делается пустым, а команда \theenumiv{•} настраивается на печать номеров источников арабскими цифрами.

В конце выполняется команда \@openbib@code. По умолчанию она пуста, но загрузка класса документа с параметром openbib изменяет ее, чтобы дополнить настройки:

```
\renewcommand\@openbib@code{ \parsep 0pt
                              \advance\leftmargin \bibindent
                              \advance\itemindent -\bibindent
                              \advance\listparinden \itemindent }.
```

В результате ее действия ширина левого поля увеличивается на длину \bibindent (см. табл. 8.4), и в соответствие с новым значением приводятся отступы \itemindent и \listparindent. Кроме этого параметр openbib изменяет команду \newblock, используемую стилями BibTEX для выделения логических блоков в описаниях источников (см. пример на с. 197). По умолчанию она вставляет перед новым блоком пробел шириной 0.11em plus 0.33em minus 0.07em, а после переопределения начинает блок с новой строки: \renewcommand\newblock{\par}.

8.10.5. Окружения quote, quotation, verse

Еще три окружения, водимые классом документа, также построены на основе списка list. Среди них самое простое устройство имеет окружение quote:

```
\newenvironment{quote}{ \begin{list}{}{ \rightmargin\leftmargin }
                                          \item\relax
                                        }{\end{list}}.
```

Метка по умолчанию в нем не устанавливается, а ширины левого и правого полей делаются равными, что дает одинаковую отбивку текста слева и справа. На входе в окружение list выполняется команда \item\relax. Команда \relax означает «ничего не делать», поэтому

текст окружения quote представляет собой абзацы одной записи, но если в нем присутствуют команды `\item[•]`, формируются новые записи. Уровень вложенности «списков» quote не должен превышать шести.

В окружении quotation добавляется отступ абзацев, и уменьшается величина зазора между ними:

```
\newenvironment{quotation}{ \begin{list}{}{
                                        \listparindent 1.5em
                                        \itemindent\listparindent
                                        \item\relax
                                        }{\end{list}}.
```

Для верстки стихов в LaTeX предусмотрено ранее не обсуждавшееся окружение verse. Оно не имеет аргументов. Верстаемое стихотворение разбивается на строфы пустыми строками, а в строфе строки разделяются командами \\. Приведем для примера пару двустиший Ренаты Мухи:

```
\begin{verse}
Один Осьминог подошел к Осьминогу\\
И в знак уваженья пожал ему ногу.

Мама — Зебра, папа — Лось.\\
Как им это удалось?
\end{verse}
```

По сравнению с quote и quotation окружение verse устроено более сложно:

```
\newenvironment{verse}{ \let \\ \@centercr
                        \begin{list}{}{
                                        \itemsep 0pt
                                        \rightmargin\leftmargin
                                        \itemindent -1.5em
                                        \listparindent\itemindent
                                        \advance\leftmargin 1.5em }
                                        \item\relax
                                         }{\end{list}}
```

Команда перевода строки заменяется в нем командой `\@centercr`, которая заканчивает строку командой `\par`, убирая при этом все пробелы, предшествующие тексту следующей строки. Таким образом каждая строка стихотворения становится абзацем.

В окружении list длина `\itemsep` устанавливается в нуль. Установкой отрицательных значений отступов `\itemindent` и `\listparindent` первые строки абзацев смещаются влево на 1.5em. Если же абзац оказывается слишком длинным, то после автоматического переноса его

вторая и последующие строки смещаются вправо относительно начала первой строки на 1.5em. Ширина отбивки слева от абзацев делается на 1.5em больше правой отбивки, но смещение первых строк выравнивает отбивки.

Так как абзацами являются стоки строфы, описанные установки приводят к тому, что ее левая граница оказывается выровненной, и только продолжения длинных строк дополнительно отбиваются слева на ширину 1.5em. Строфа одинаково отбивается слева и справа, но ее правая граница не выравнивается.

8.10.6. Окружение trivlist и его клоны

Самостоятельной модификацией списка list является окружение trivlist. Оно не имеет аргументов и использует настройки списков, описанные в начале раздела. Контроль уровня вложенности в нем отсутствует, команды \list• не выполняются, поэтому настройки всех окружений одинаковы. Длины \labelwidth, \leftmargin и \itemindent устанавливаются в нуль, а длина \parsep приравнивается к \parskip. Строки внутри и вне окружения имеют одинаковую длину, абзацы — одинаковую вертикальную отбивку.

Окружения center, flushleft и flushright, выравнивающие текст, строятся на основе trivlist:

```
\newenvironment{center}{
      \begin{trivlist} \centering   \item\relax }{\end{trivlist} },
\newenvironment{flushleft}{
      \begin{trivlist} \raggedright \item\relax }{\end{trivlist} },
\newenvironment{flushright}{
      \begin{trivlist} \raggedleft  \item\relax }{\end{trivlist} }.
```

Оно также является основой окружений verbatim и verbatim*:

```
\newenvironment{verbatim}{
       \begin{trivlist} \item\relax  Преамбула }{\end{trivlist} },
\newenvironment{verbatim*}{
       \begin{trivlist} \item\relax  Преамбула }{\end{trivlist} }.
```

Преамбула содержит дополнительный набор команд для буквального воспроизведения текста.

Среди математических конструкций окружение trivlist используют теоремы \newtheorem{•}{•}[•] и \newtheorem{•}[•]{•}.Их код имеет сложную структуру и потому здесь не анализируется.

Глава 9

Формулы

В гл. 4 описаны основные средства верстки формул. Здесь мы обсудим настройки, регулирующие их встраивание в текст, и редко используемые ресурсы коллекции пакетов $\mathcal{AMS}$-LaTeX. Высокая стабильность данной коллекции обеспечивается тем, что ее развитие давно заморожено. В качестве расширения разработчики $\mathcal{AMS}$-LaTeX рекомендуют использовать пакет mathtools [31], который пока не является таким же стандартом, что и сам $\mathcal{AMS}$-LaTeX, но его разработка ведется как официальная часть проекта LaTeX3.

В данной главе мы рассмотрим взаимодополняющие ресурсы и настройки $\mathcal{AMS}$-LaTeX и mathtools, а также настройки некоторых конструкций, уже обсуждавшихся ранее.

9.1. Пакеты коллекции $\mathcal{AMS}$-LaTeX

Ранее пакеты коллекции $\mathcal{AMS}$-LaTeX обсуждались по отдельности, здесь мы приведем их сводку. Коллекцию составляют классы документов amsart, amsbook и amsproc [70] Американского математического общества и пакеты amsmath, amssymb, amsbsy, amsfonts, eucal, amscd, amsopn, amstext, amsthm, amsmidx. Для доступа к основным ресурсам коллекции достаточно загрузить пакеты amsmath и amssymb, так как первый автоматически загружает amsbsy, amsopn, amstext, а последний — amsfonts. Перечислим кратко назначение пакетов.

Пакеты amssymb, amsfonts и eucal вводят большое количество символов, а также шрифты `\mathbb`, `\mathfrak` и `\mathscr`. По умолча-

нию рукописный шрифт `\mathscr` заменяет каллиграфический шрифт `\mathcal`, а чтобы пользоваться обоими шрифтами, пакет eucal нужно загружать с параметром mathscr.

Пакеты amscd и theorem обеспечивают верстку коммутативных диаграмм и теорем.

Пакет amstext определяет команду `\text`.

Пакет amsopn расширяет набор математических функций и вводит команды `\operatorname`, `\DeclareMathOperator`.

Пакет amsbsy вводит команду `\boldsymbol` для печати жирных символов, а также команду `\pmb{`*выражение*`}`, которой можно воспользоваться, если они в шрифтах отсутствуют. Данная команда генерирует их путем наложения друг на друга обычных символов, печатаемых с небольшим смещением.

Пока не упоминавшийся пакет amsmidx [71] обеспечивает возможность формирования нескольких указателей (см. разд. 11). Для сбора записей он определяет команду `\index{`*имя файла*`}{`*запись*`}`, а для печати — команду `\Printindex{`*имя файла*`}{`*заголовок указателя*`}`. Первым аргументом обеих команд является имя файла, в который заносятся данные указателя. Еще одна команда `\indexcomment{`*текст*`}`, которая ставится перед `\Printindex`, позволяет вставить под заголовком указателя комментарий.

Пакет amsmath определяет большое число математических конструкций, а также окружения для создания блоков и нумерованных формул. Его настройки управляют оформлением некоторых элементов формул.

9.2. Настройка формул

9.2.1. Настройки пакета amsmath

Основным пакетом коллекции $\mathcal{AMS}$-LaTeX является amsmath, поэтому рассмотрим сначала его настройки, управляющие оформлением некоторых элементов формул. Приведенные ниже параметры перечисляются списком через запятую при его загрузке. Настройки, используемые по умолчанию, выделены жирным шрифтом.

alignedleftspaceyes | alignedleftspaceno | **alignedleftspaceyesifneg** — регулировка тонкого пробела `\,` перед блоками aligned, alignedat и gathered. Первые два параметра, соответственно, вставляют и *не* вставляют его, а третий вставляет пробел лишь в том случае, если перед блоками находится отрицательный пробел.

centertags | tbtags — размещение номера уравнения, содержащего блок split . Параметр centertags центрирует номер по высоте блока. При использовании tbtags номер проставляется в последней строке выражения, верстаемого блоком, если уравнение нумеруется справа, или в первой строке, если оно нумеруется слева.

leqno | **reqno** — вывод номера слева или справа от формулы.

fleqn — выравнивание формул по левому краю.

intlimits | **nointlimits** — размещение пределов у интегралов в вынесенных формулах. Параметр intlimits расставляет пределы сверху и снизу, а nointlimits — сбоку от знака интеграла.

sumlimits | nosumlimits — аналогичное регулирование пределов остальных больших операторов (см. разд. 4.1.3).

namelimits | nonamelimits — аналогичное регулирование пределов функций, перечисленных в табл. 4.7 и В.20.

Параметры leqno, reqno и fleqn являются аналогами параметров стандартных классов, и чтобы получить одинаковое оформление номеров формул окружениям equation, eqnarray и окружениями пакета amsmath, их лучше устанавливать при загрузке класса документа. Например, при загрузке пакета amsmath с параметром fleqn формулы, верстаемые окружением eqnarray, останутся центрированными, тогда как при загрузке класса с тем же параметром они прижмутся к левой границе текста.

9.2.2. Настройки пакета mathtools

Пакет mathtools [31] сам загружает пакет amsmath. Дублировать загрузку не следует, а настройки amsmath можно перечислить в списке его параметров. Они будут автоматически переданы пакету amsmath. Параметры загрузки самого пакета mathtools немногочисленны:

fixamsmath | donotfixamsmath — коррекция ошибок, обнаруженных в пакете amsmath.

allowspaces | **disallowspaces** — запрет пробелов, разделяющих аргументы и параметры окружений.

Используемые по умолчанию настройки, выделенные жирным шрифтом, лучше не менять, так как они устраняют возможность некорректного действия некоторых конструкций пакета amsmath, которые разработчики $\mathcal{AMS}$-LaTeX, видимо, в дальнейшем исправят сами.

Для настройки конструкции пакета mathtools предназначена декларация `\mathtoolsset{`*список настроек*`}`. Ее аргументом является *список настроек*, определенных для различных операций, например `\mathtoolsset{mathic, multlined-pos=t, showonlyrefs}`. Многие настройки принимают значения true и false. Используя их, false нужно указывать обязательно, а true можно опустить, ограничившись внесением настройки в список. Таким образом, настройки mathic=true и mathic эквивалентны, а обратным им является mathic=false. Данная настройка корректирует отбивку строчных формул в тексте, набранном курсивом. По умолчанию используется ее значение false, но лучше заменить его на true.

Список настроек, применяемых для верстки всего документа, следует поместить в преамбуле, а выполнение отдельных операций можно корректировать декларациями `\mathtoolsset`, размещенными в рукописи. Поясним сказанное примером. Настройка original-intertext=false уменьшает вертикальную отбивку текста, вставляемого между строк формулы командой `\intertext`, которую пакет amsmath задает слишком большой. Она действует по умолчанию и в документе в целом лучше использовать ее. Если же в какой-то формуле понадобится большая отбивка, разместите ее между декларациями

```
\mathtoolsset{original-intertext}
    формула
\mathtoolsset{original-intertext=false}.
```

Первая увеличит отбивку, а вторая вновь уменьшит ее. Можно ограничиться и лишь первой декларацией, заключив ее вместе с формулой в группирующие скобки, тогда при выходе из группы уменьшенная величина отбивки восстановится автоматически.

Перечислим сначала настройки уже обсуждавшихся конструкций, опущенные для простоты изложения в гл. 4. Значения настроек, используемые по умолчанию, выделены жирным шрифтом.

Двоеточие

centercolon = true | **false**

Вводя многочисленные символы с двоеточием, представленные в табл. 4.5, пакет mathtools позволяет настроить и вывод самого двоеточия. По умолчанию его нижняя точка находится в строке, а параметр centercolon центрирует двоеточие относительно ее осевой линии. Независимо от текущих настроек, команды `\vcentcolon` и `\ordinarycolon` печатают центрированное и обычное двоеточие.

Маленькие матрицы

smallmatrix-align = **c** | l | r,
smallmatrix-inner-space = *пробел*.

Окружения, верстающие маленькие матрицы (см. с. 105), имеют параметр, регулирующий выравнивание столбцов, которые по умолчанию центрируются. Настройка smallmatrix-align позволяет установить для них левое (l) или правое (r) выравнивание или снова вернуть центрирование (c).

Настройка smallmatrix-inner-space задает ширину пробела, разделяющего столбцы. Ее значениями являются команды пробелов, взамен используемого по умолчанию тонкого пробела `\,`.

Блок multlined

multlined-pos = b | **c** | t, firstline-afterskip = *длина*,
multlined-width = *длина*, lastline-preskip = *длина*,

Параметр multlined-pos, позволяет заменить проводимое по умолчанию центрирование верстаемого выражения по высоте относительно осевой линии формулы (c) на встраивание его верхней (t) или нижней (b) строки в строку формулы. Остальные параметры, по умолчанию равные нулю, регулируют ширину верстаемого выражения. Вычисленная в ходе верстки длина первой строки увеличивается на длину, заданную параметром firstline-afterskip. К длине последней строки добавляется значение параметра lastline-preskip. Скорректированные таким образом длины используются в компоновке выражения в целом.[1] Отметим, что обсуждавшееся на с. 108 окружение multlined можно настроить, изменяя его собственные параметры непосредственно при использовании, и как показывает опыт, зачастую это более удобно, чем корректировать общие настройки.

Параметр multlined-width позволяет задать ширину выражения в целом. Она будет использована, только если окажется больше ширины, определенной в ходе верстки. В этом случае первая и последняя строки раздвинутся до заданного размера. В противном же случае сверстанное выражение не изменится.

Покажем действие данных настроек на примере трех формул, первая из которых сверстана без настроек, а настройки других указаны сверху:

[1] В отдельном блоке multlined аналогичное действие производят пробелы `\hspace{•}`, поставленные вручную в конце первой или начале последней строки выражения.

$$a+b+c+ \\ +d+e=x,$$

multlined-width = 9.3em

$$a+b+c+ \\ +d+e=x,$$

lastline-preskip = 2.11em

$$a+b+c+ \\ +e+d=x.$$

9.3. Отбивка формул

Ядро LaTeX определяет ряд длин, регулирующих отбивку вынесенных формул. Часть их значений устанавливает класс документа.

`\jot` (3pt) — высота дополнительного пробела между строками в формулах и блоках, добавляемая к вычисляемым интервалам.

`\mathindent` — расстояние от левой границы текста до формул, выравниваемых с помощью параметра fleqn. По умолчанию приравнивается длине `\leftmargini`.

`\abovedisplayskip` и `\belowdisplayskip` — высота отбивок над и под формулами, перекрывающимися с предшествующей строкой текста.

`\abovedisplayshortskip` и `\belowdisplayshortskip` — высота отбивок над и под формулами, не перекрывающимися с предшествующей строкой текста.

В скобках указаны значения, действующие в стандартных классах. Величины остальных длин зависят от размера основного шрифта документа.

9.3.1. Вертикальные отбивки

Если формула не умещается на странице, изменив длину `\jot`, можно слегка уменьшить ее высоту, как показано в следующем примере:

```
\begin{multline*}
  a+b+c+d+e+\\
    +f+g+h+i=0.
\end{multline*}
```

$$a+b+c+d+e+ \\ +f+g+h+i=0.$$

```
{ \setlength\jot{0pt}
\begin{multline*}
  ...
\end{multline*} }
```

$$a+b+c+d+e+ \\ +f+g+h+i=0.$$

Группирующие скобки, ограничивающие область изменения значения `\jot` в предыдущем примере, легко нарушить, поэтому пакет mathtools вводит окружение spreadlines :

\begin{spreadlines}{длина}.

Его аргументом является значение \jot, применяемое к формулам внутри окружения, например:

```
\begin{spreadlines}{0pt}
  \begin{multline*}
     ...
\end{spreadlines}
```

$$a+b+c+d+e+ \\ +f+g+h+i=0.$$

Изменение длины \jot меняет отбивку *всех* строк формул, не влияя на межстрочный интервал внутри блоков. Если же нужно изменить расстояние между строками блока или частью строк формулы, воспользуйтесь параметром команды перевода строки:

```
$ \begin{multlined}[c][9.4em]
  a+b+c+d+ \\[-0.5ex]
  +g+h+i=0.
\end{multlined} $
```

$$a+b+c+d+ \\ +g+h+i=0.$$

Размеры промежутков между текстом и вынесенными формулами (кроме eqnarray) задают длины \abovedisplayskip, \belowdisplayskip и \abovedisplayshortskip, \belowdisplayshortskip. Первая пара, регулирующая отбивку формул, перекрывающихся с предшествующим текстом, имеет бóльший размер. Изменение этих длин регулирует высоту отбивок всех формул. Если же требуется настроить их для одной формулы, поставьте до или после[2] нее пробел \vspace. Напомним, что положительная длина в аргументе данной команды отодвинет формулу от текста, а отрицательная — уменьшит зазор между ними.

В зависимости от заполненности страницы верхняя отбивка формул, верстаемых окружением equation, может уменьшаться. У окружений пакета amsmath такое поведение не предусмотрено, поэтому пакет mathtools вводит для них команду \SwapAboveDisplaySkip, используемую непосредственно в формуле. Для верхней отбивки она использует длину \abovedisplayshortskip, а нижнюю отбивку не меняет. Данная команда должна стоять в формуле самой первой, другие команды и символы, оказавшись перед ней, вызовут ошибку компиляции. Вот как выглядит результат ее действия:

[2] Команду \vspace должна отделять от формулы (или текста) пустая строка, без которой изменится расстояние между первой и последующими строками текста, а не зазор между ним и формулой.

```
\begin{gather}
  x=y
\end{gather}
```

$$x = y \tag{9.1}$$

```
\begin{gather}
  \SwapAboveDisplaySkip
   x=y
\end{gather}
```

$$x = y \tag{9.2}$$

Горизонтальными линиями показаны границы текста.

Отбивка выражений

Ранее говорилось, что вертикальные отбивки выражений можно корректировать распорками `\vphantom` и `\mathstrut`. С этой целью также можно нарисовать фиктивную линию, имеющую нулевую ширину `\rule[`*глубина*`]{0pt}{`*высота*`}`, высота и глубина которой учитываются в вычислениях размеров. Пакет mathtools расширяет эти возможности командой `\xmathstrut[`*глубина*`]{`*высота*`}`, вставляющей распорку регулируемого размера. С нулевым аргументом и без параметра она эквивалентна стандартной команде `\mathstrut`, вставляющей распорку с глубиной и высотой круглой скобки, а параметр и аргумент позволяют добавить доли ее размеров. Положительные значения увеличивают размеры, а отрицательные их уменьшают. Если параметр опущен, высота и глубина изменяются одинаково.

Поясним ее действие на примерах. Сформируем формулу из двух букв x и поставим между ними распорку, окружив ее для наглядности пробелами и сделав видимой. Для этого поместим распорку в рамку `\boxed`, убрав пустые поля, чтобы преобразовать рамку в вертикальную линию с высотой и глубиной распорки:

```
{\fboxsep=0pt $x \boxed{\xmathstrut[•]{•}} x$},
```

$$x \boxed{\xmathstrut[\bullet]{\bullet}} x \quad , \quad x \boxed{\xmathstrut[.25]{\bullet}} x \quad , \quad x \boxed{\xmathstrut[-.25]{\bullet}} x \quad , \quad x \boxed{\xmathstrut[-1]{0}} x \quad , \quad x \boxed{\xmathstrut[0]{-1}} x \, .$$

```
{0}      {.25}     {-.25}     [-1]{0}     [0]{-1}
```

В примерах внизу указаны значения аргументов и параметров команд `\xmathstrut`, а над ними выведены сверстанные формулы. Первой показана распорка обычного размера. В двух следующих примерах ее высота и глубина увеличиваются и уменьшаются на 25 %. В двух последних примерах, чтобы обнулить один из размеров, значение параметра или аргумента устанавливается равным минус единице.

9.3.2. Горизонтальные отбивки

Корректировать горизонтальную отбивку вынесенных формул приходится нечасто. Левая отбивка отделяет их от границы текста, а правая — от номеров, и наоборот, если формулы нумеруются справа. По умолчанию левая и правая отбивка одинаковы, поэтому формула центрируется (без учета ширины номера) относительно границ колонки текста или страницы. При выравнивании слева ширину сдвига формул от границы текста задает длина `\mathindent`, изменение которой настраивает левую отбивку *всех* формул.

Напомним, что регулировать пробелы в формулах позволяют команды `\hspace{•}`и `\mspace{•}`, а также горизонтальные фантомы `\hphantom{•}`. Вставляя их в начале и конце строки, можно сдвигать выражение относительно границ и номера формулы.

Чтобы кардинально изменить отбивку формулы, воспользуйтесь следующим методом. Поместите ее в министраницу и подберите ширину последней так, чтобы она обеспечила требуемый пробел между выражением и номером. Например, так будет выглядеть короткая формула на министранице шириной 0.2`\linewidth`:[3]

```
\begin{minipage}{.2\linewidth}
  \begin{gather} x=y. \end{gather}
\end{minipage}
```

$$x = y. \quad (9.3)$$

Здесь и далее горизонтальными линиями показана ширина министраниц. Министраница является частью текста и ее положение можно регулировать, используя пружины и пробелы, как описано в разд. 2.2.3. Воспользовавшись жесткой пружиной, сдвинем нашу формулу на левый край страницы:

```
\begin{minipage} ... \end{minipage}\hfill\mbox{}
```

$$x = y \quad (9.3)$$

Описанный способ применим при любом типе выравнивания формул.

Окружение multline имеет собственную настройку горизонтальных отбивок. Длины `\multlinegap` и `\multlinetaggap` задают, соответственно, величину отступа первой строки от левой границы текста и ширину зазора между концом последней строки и номером формулы. По умолчанию обе они имеют величину, равную 10pt, дающую следующие отбивки

[3] Сравните полученную формулу с точно такими же формулами (9.2) и (9.1), размещенными в министраницах шириной 0.45`\linewidth`.

```
\begin{multline}
  a+b+c+d+ \\
  +e+f=0,
\end{multline}
```

$$a+b+c+d+ \\ +e+f=0, \quad (9.4)$$

Удвоив длины, получим следующий результат:

```
\setlength\multlinegap{20pt}
\setlength\multlinetaggap{20pt}
\begin{multline}
   ...
```

$$a+b+c+d+ \\ +e+f=0. \quad (9.5)$$

Для коррекции отбивок выражений, связанных между собой общими вертикалями выравнивания в уравнениях типа align и eqnarray и блоках aligned, alignedat, split и cases, пакет mathtools вводит команду `\MoveEqLeft[число]`. В ее параметре указывается действительное число, задающее величину сдвига выражения в единицах em. Данная команда должна стоять в начале выражения, в котором нужно убрать знак `&`. При этом его связь с вертикалью выравнивания теряется и оно сдвигается влево, если параметр положителен, или вправо, если он отрицателен. По умолчанию выражение сдвигается влево на 2em.

Разберем действие команды `\MoveEqLeft` на следующем примере:

```
\begin{align}
  x &= a+b+c+d+e,\\
  x+y &= d+e+f,    \\
  x+y+z &= f+g+h.
\end{align}
```

$$x = a+b+c+d+e, \quad (9.6)$$
$$x+y = d+e+f, \quad (9.7)$$
$$x+y+z = f+g+h. \quad (9.8)$$

```
\begin{align}
  \MoveEqLeft[4]
  x = a+b+c+d+e,\\
  x+y &= d+e+f,   \\
  x+y+z &= f+g+h.
\end{align}
```

$$x = a+b+c+d+e, \quad (9.9)$$
$$x+y = d+e+f, \quad (9.10)$$
$$x+y+z = f+g+h. \quad (9.11)$$

Слева показана конструкция align без коррекции. Первое уравнение имеет длинную правую часть, а последнее — левую. Как следствие, первое уравнение, согласованное с общей вертикалью выравнивания, оказавшись слишком длинным, сталкивает свой номер вниз. В конструкции справа команда `\MoveEqLeft` отключает его от вертикали выравнивания и сдвигает его влево на 4em. Этого вполне достаточно, чтобы уравнение уместилось в строке вместе с номером.

Покажем без комментариев коррекцию еще одной конструкции:

```
\begin{align}
  x &= a+b, & x^2 &= y+z,\\
  y &= c+d, & y^2 &= z,  \\
  z &= e+f, & \MoveEqLeft[1]
              x^2-y^2 = a.
\end{align}
```

$$x = a + b, \quad x^2 = y + z, \tag{9.12}$$
$$y = c + d, \quad y^2 = z, \tag{9.13}$$
$$z = e + f, \quad x^2 - y^2 = a. \tag{9.14}$$

9.4. Математические боксы

Боксы \mbox, \makebox, \fbox и \framebox (см. разд. 7.3) в формулах вставляют текст, а для создания математических боксов имеются свои аналогичные команды:

`\mathmbox{`*выражение*`}`, `\boxed{`*выражение*`}`,

`\mathmakebox[`*ширина*`][`*выравнивание*`]{`*выражение*`}`.

Их аргументом являются математические выражения, которые должны быть полностью определены на момент создания бокса. Они могут содержать блоки (целиком), но не должны содержать их частей, равно как и частей выражений, разделенных символами & или командами \left и \right. Команды\mathmbox и \mathmakebox, введенные пакетом mathtools, создают боксы без рамок. Для боксов в рамке пакет amsmath определяет команду \boxed. В таких боксах выражения верстаются размером \displaystyle. Это хорошо видно из примера, в котором левая и правая дроби верстаются в стиле строчной и вынесенной формулы:

```
$\frac{a+b}{c+d}=\boxed{\frac{a+b}{c+d}}$
```
$\rightsquigarrow$ $\frac{a+b}{c+d} = \boxed{\frac{a+b}{c+d}}$

Как и в текстовых боксах, параметры рамки устанавливают длины \fboxrule и \fboxsep.

Команда \boxed не может обвести рамкой выражения, находящиеся в разных ячейках математических конструкций. С этой целью пакет mathtools вводит команду `\Aboxed{`*левая часть & правая часть*`}`. Она предназначена для использования в блоках split, cases, а также блоках и уравнениях типа align. Ее аргумент должен содержать символ &, связывающий части выражения с общей вертикалью выравнивания:

```
\begin{align*}
  z+ \Aboxed{x &= y} +a,\\
  \Aboxed{z+x &= y+a.}
\end{align*}
```

$$z+\boxed{x = y}+a,$$
$$\boxed{z+x = y+a.}$$

С помощью команды `\Aboxed` не удастся заключить в рамку выражения, находящиеся более чем в двух ячейках или столбцах матриц. Ячейки матриц будут объединены, заключены в рамку и все вместе помещены в правый столбец, а в уравнениях и блоках выражения третьей и последующих ячеек бесследно исчезнут.

Как говорилось на с. 101, в ядре LaTeX определен математический бокс `\lefteqn{•}`, имеющий нулевую ширину. Пакет mathtools добавляет к нему еще три, действующие аналогично текстовым боксам нулевой ширины, обсуждавшимся на с. 169:

Команды	Примеры		
`\mathllap{`выражение`}`	`*\mathllap{\_}*`	⇝	_**,
`\mathrlap{`выражение`}`	`*\mathrlap{\_}*`	⇝	**_,
`\mathclap{`выражение`}`	`*\mathclap{\_}*`	⇝	_**_.

С их помощью можно синтезировать экзотические символы, к примеру `:\mathclap-:` ⇝ :—: или `\}\mathllap\{` ⇝ }{.

Бокс `\lefteqn` является аналогом бокса `\mathrlap`, отличаясь от него тем, что верстает выражение размером `\dispalystyle`, тогда как `\mathrlap` согласует размер выражения с его положением в формуле.

Команды `\lefteqn`, `\mathllap` и `\mathrlap` позволяют добавить в начало или конец формулы выражения, длины которых не будут учитываться. Это может оказаться полезным, чтобы «обмануть» компилятор, когда слишком длинное выражение выталкивает номер из строки. Наиболее часто боксы нулевой длины применяются для расстановки индексов.

9.5. Пределы и индексы

Целый ряд конструкций и настроек пакета mathtools управляет версткой пределов и индексов.

Громоздкие пределы больших операторов делают выражение «рыхлым», избавиться от вносимых ими избыточных пробелов помогают боксы нулевой длины. Сравните:

```
X= \sum_{1<i<k<100} x_{ik}
```
⇝ $X = \sum_{1<i<k<100} x_{ik},$

```
X= \sum_{\mathclap{...} x_{ik}
```
⇝ $X = \sum_{1\le i\le k\le 100} x_{ik},$

```
X= \sum_{\mathllap{...}} x_{ik}
```
$$\rightsquigarrow \quad X = \sum_{1\le i\le k\le 100} x_{ik},$$
```
X= \sum_{\mathrlap{...}} x_{ik}
```
$$\rightsquigarrow \quad X = \sum_{1\le i\le k\le 100} x_{ik}.$$

В первом примере использовано стандартное оформление пределов суммирования, а в остальных применены обсуждавшиеся ранее боксы. Среди них только `\mathclap` дает приемлемый результат, тогда как позиции пределов, сверстанных командами `\mathllap` и `\mathrlap`, смещаются налево или направо. Эти боксы могут быть полезны в других случаях, например для корректировки пределов интегралов:

```
\int\limits_{-\infty}^\infty x\,dx=0
```
$$\rightsquigarrow \quad \int\limits_{-\infty}^{\infty} x\,dx = 0,$$
```
\int...^{\mathrlap\infty} x\,dx=0
```
$$\rightsquigarrow \quad \int\limits_{-\infty}^{\infty} x\,dx = 0.$$

В стандартной расстановке, показанной в первом примере, верхний предел выглядит сдвинутым влево, поэтому команда `\mathrlap`, примененная во втором примере, «возвращает его на место». Отметим, что тонкий пробел «`\,`» хорошо отделяет дифференциал от интегрируемого выражения.

Для гибкого выравнивания пределов больших операторов пакет mathtools вводит команду

`\smashoperator[`*выравнивание*`]{`*оператор с пределами*`}`.

По умолчанию она, аналогично команде `\mathclap`, убирает пустые поля вокруг оператора, а параметры `l` и `r` позволят оставить одно из них: `l` убирает левую отбивку, `r` — правую. Совместное использование параметров (`lr` или `rl`) эквивалентно действию по умолчанию, т. е. отсутствию пустых полей. Покажем действие команды на примере использованного ранее выражения:

```
X=\smashoperator{\sum_{1< i< k< 100}} x_{ik}
```
$$\rightsquigarrow \quad X = \sum_{1<i<k<100} x_{ik}\,,$$
```
X=\smashoperator[l]{...} x_{ik}
```
$$\rightsquigarrow \quad X = \sum_{1<i<k<100} x_{ik}\,,$$
```
X=\smashoperator[r]{...} x_{ik}
```
$$\rightsquigarrow \quad X = \sum_{1<i<k<100} x_{ik}\,.$$

Еще одна конструкция пакета mathtools

`\adjustlimits` *оператор 1* *предел 1* *оператор 2* *предел 2*

решает проблему выравнивания пределов двух соседних операторов или функций, когда они оказываются на разной высоте, например $\lim\limits_{n\to0}\sup\limits_{p^2>nK}$. Вот как выглядит результат ее действия:

```
\adjustlimits\lim_{n\to0}\sup_{p^2>nK}
```
$\rightsquigarrow$ $\adjustlimits\lim_{n\to0}\sup_{p^2>nK}$.

Конструкция `\adjustlimits` верстает пределы в режиме `\limits`. Ее устройство предполагает, что за командой следуют два оператора (или функции, см. табл. 4.7), каждый из которых имеет один предел, при этом ни операторы, ни пределы в дополнительные скобки заключать не следует, иначе конструкция будет работать неправильно. Пределы должны быть одного типа, верхние или нижние, но не оба, однако выравниваются только нижние.

В подкоренных выражениях, числителях и знаменателях дробей, пределах операторов и некоторых других конструкциях верхние индексы печатаются более компактно, нежели в выражениях, стоящих в основной строке формулы. Пакет mathtools позволяет использовать такую компактную верстку в любой части формулы. Для этого он вводит команду `\cramped[`*стиль*`]{`*выражение*`}`. Сравните, например, обычную и компактную печать степени в строчной формуле:

```
$x^{2^3}=x^8,\ \scriptstyle x^{2^3}=x^8$
```
$\rightsquigarrow$ $x^{2^3}=x^8,\ \scriptstyle x^{2^3}=x^8$,

```
$\cramped{...},\ \cramped[\scriptstyle]{...}$
```
$\rightsquigarrow$ $x^{2^3}=x^8,\ \cramped{...},\ \cramped[\scriptstyle]{...}$.

В параметре команды `\cramped` можно задать математический *стиль*, в котором будет верстаться *выражение* (см. разд. 4.2).

Для верстки компактных индексов, помещенных в боксы нулевой ширины, определены специальные команды

```
\crampedllap[стиль]{выражение},
\crampedrlap[стиль]{выражение},
\crampedclap[стиль]{выражение}.
```

Чтобы сократить время компиляции, их рекомендуется использовать вместо последовательности команд `\math`•`lap{\cramped{...}}`. Для сравнения приведем пару выражений:

$$X=\sum_{i^2,k^2<100} x_i^k, \qquad X=\sum_{i^2,k^2<100} x_i^k.$$

В первом переменная x_i^k набрана обычным образом, а предел суммы верстает команда `\mathclap`. Второе создано конструкцией

```
X= \sum_{\crampedclap{i^2,k^2< 100}} \cramped{x_i^k}.
```

Для верстки нескольких строк компактных индексов пакет mathtools определяет аналоги ранее обсуждавшейся конструкции `\substack{•}` и окружения `subarray` (см. разд. 4.1.3)

`\crampedsubstack{`*индексы* `\\` *индексы* `\\ ...}`,
`\begin{crampedsubarray}{`*выравнивание*`}`.

Аргумент окружения `crampedsubarray` задает выравнивание строк слева (`l`), справа (`r`) или по центру (`c`).

На с. 93 описана конструкция `\prescript{•}{•}{•}`, верстающая левые индексы. Для их печати можно задать шрифты:

prescript-sub-format = *алфавит*,
prescript-sup-format = *алфавит*,
prescript-arg-format = *алфавит*.

В настройках указываются либо команды математических алфавитов, либо команды `\text` или `\bm`. Настроим с их помощью команду, печатающую изотопы химических элементов:

```
\newcommand\isotope[2]{\begingroup\mathtoolsset{
    prescript-sup-format = \text, prescript-arg-format = \text}%
  \ensuremath{\prescript{#1}{}{#2}}\endgroup}
```

Аргументами созданной команды `\isotope` являются масса и обозначение элемента. Наименование изотопа верстается в виде формулы, создаваемой командой `\ensuremath`. Чтобы оно не выпадало из окружающего текста, применены команды `\text`, использующие для печати текстовый шрифт. Команды `\begingroup` и `\endgroup` создают группу, ограничивающую область действия настроек. Знак комментария в конце второй строки предотвращает появление лишних пробелов.

Используем созданную команду для печати изотопов водорода ^{1}H, ^{2}H, $^{\mathbf{3}}\mathbf{H}$, выделив радиоактивный жирным шрифтом:

```
.... \isotope{1}{H}, \isotope{2}{H}, \textbf{\isotope{3}{H}} ...
```

9.6. Парные скобки

Пакет mathtools вводит ряд деклараций, оптимизирующих работу с парными скобками. Они позволяют автору создавать собственные команды для нужных ему конструкций. Декларация

`\DeclarePairedDelimiter{`*команда*`}{`*левый разд.*`}{`*правый разд.*`}`,

определяет команду, имеющую аргумент и параметр. Ее аргументом является выражение, заключаемое в скобки, а в параметре можно указать команду, регулирующую их размер (см. разд. 4.1.4).

Создадим для примера функцию модуля:

```
\DeclarePairedDelimiter\abs\lvert\rvert.
```

В простейшем случае используем ее без параметра: `\abs{x}` $\rightsquigarrow |x|$. Теперь возьмем выражение с высотой, большей чем у обычного символа, и подберем к нему размер «скобок»: `\abs[\big]{\frac{a}{b}}` $\rightsquigarrow \left|\frac{a}{b}\right|$.

Пользоваться скобками, имеющими фиксированный размер, не очень удобно. При копировании созданной функции из строчной в вынесенную формулу скобки придется увеличить: `\abs[\bigg]{\frac{a}{b}}` $\rightsquigarrow \left|\frac{a}{b}\right|$. Кроме того, размеры выражения могут оказаться столь большими, что максимальной фиксированной высоты скобок может и не хватить, поэтому декларация `\DeclarePairedDelimiter` автоматически создает еще и команду, имя которой заканчивается звездочкой, а размер скобок регулирует конструкция `\left`•`...\right`•. Она имеет только аргумент. Напечатаем с помощью такой команды, неявно созданной в нашем примере, громоздкое выражение:

```
\abs*{\sqrt{\frac{f(a)}{f(b)}}}
```
$\rightsquigarrow \left|\sqrt{\frac{f(a)}{f(b)}}\right|.$

Если выражению, охватываемому скобками, требуются свои аргументы, их обеспечивает декларация

```
\DeclarePairedDelimiterX{команда}[N]{лев. разд.}{прав. разд.}{код}.
```

Ее параметр N задает число аргументов, которое будет иметь новая команда, а в последний аргумент записывается код выражения, охватываемого скобками. Как и в предыдущем случае, она определяет команду с параметром, в котором указывается размер скобок, и команду «со звездочкой», параметра не имеющую. Применим `\DeclarePairedDelimiterX` для создания волновой функции запутанных состояний ψ_1 и ψ_2:

```
\DeclarePairedDelimiterX\entangled[2]{\lvert}{\rangle}%
   {\begin{matrix} #1 \\ #2 \end{matrix}}
```

Созданная команда `\entangled` имеет два аргумента, из которых формируется столбец матрицы matrix, заключенный между знаком модуля и угловой скобкой. Запишем с ее помощью определение волновой функции:

```
\entangled*{\psi_1}{\psi_2} =
```

```
\frac{1}{\sqrt2} [\psi_1(x_1)\psi_2(x_2) -\psi_1(x_2)\psi_2(x_1)].
```

$$\left|\begin{matrix}\psi_1\\ \psi_2\end{matrix}\right\rangle = \frac{1}{\sqrt{2}}[\psi_1(x_1)\psi_2(x_2) - \psi_1(x_2)\psi_2(x_1)].$$

Если выражение содержит скобки, размер которых нужно согласовать со скобками, охватывающими само выражение, воспользуйтесь командой `\delimsize`. В нее записывается размер скобок, указываемый в параметре создаваемой команды. Поясним сказанное, определив команду `\Entangled`, верстающую матричный элемент волновой функции запутанных состояний:

```
\DeclarePairedDelimiterX\Entangled[3]{\langle}{\rangle}%
  { \begin{smallmatrix} #1 \\ #2 \end{smallmatrix}
  \delimsize| #3 \delimsize|
  \begin{smallmatrix} #1 \\ #2 \end{smallmatrix} }
```

`\Entangled[\big]{\psi_1}{\psi_2}{\hat H}` $\rightsquigarrow$ $\left\langle \begin{smallmatrix}\psi_1\\ \psi_2\end{smallmatrix} \middle| \hat{H} \middle| \begin{smallmatrix}\psi_1\\ \psi_2\end{smallmatrix} \right\rangle$.

В отличие от предыдущего случая, для формирования волновой функции мы использовали окружение smallmatrix и добавили команде `\Entangled` третий аргумент для ввода оператора. Оператор отделяется от волновой функции знаками модуля, высоту которых команды `\delimsize` согласуют с размером угловых скобок.

Самые сложные конструкции создает декларация

`\DeclarePairedDelimiterXPP{`*команда*`}[N]`
`{`*код до*`}{`*лев. разд.*`}{`*прав. разд.*`}{`*код после*`}{`*код команды*`}`.

С помощью второго и пятого аргументов в ней добавлена возможность выполнить действия *до* и *после* печати охватывающих скобок. Такая возможность представляется избыточной, так как это можно сделать и без участия создаваемой команды. Мы не нашли ей достойного применения, заинтересованный читатель может сделать это самостоятельно.

9.7. `\intertext` и другие связки

Для вставки текста между строками формул пакет amsmath определяет команду `\intertext{•}` (см. разд. 4.5), которая отбивает текст сверху и снизу промежутками высотой `\abovedisplayskip` и `\belowdisplayskip`. Обычно они выглядят чрезмерно большими, поэтому пакет mathtools уменьшает их и дополнительно вводит команду с еще меньшей отбивкой: `\shortintertext{`*текст*`}`. Сравните:

$$a+b+c+ \quad +e+f+g,$$

\intertext amsmath

$$a+b+c+ \quad +e+f+g,$$

\intertext mathtools

$$a+b+c+ \quad +e+f+g.$$

\shortintertext

Слева и в центре показано действие команды \intertext с отбивками пакетов amsmath и mathtools, а справа — \shortintertext. Параметры

above-intertext-sep = длина (0pt), above-shortintertext-sep = длина (3pt),
below-intertext-sep = длина (0pt), below-shortintertext-sep = длина (3pt)

позволяют автору настроить отбивки по своему усмотрению. По умолчанию изменение межстрочного интервала \jot в окружении spreadlines не затрагивает отбивку вставляемого текста, но параметры

original-intertext = true | **false**, original-shortintertext = true | **false**

корректируют это поведение. Если установить их в значение true, отбивки тоже будут увеличены на длину \jot.

Пакет mathtools позволяет связать стрелками строки формул, верстаемых окружениями типа alignat, для этого между ними нужно поставить команду

\ArrowBetweenLines[символ] , или \ArrowBetweenLines*[символ] .

По умолчанию вставляется стрелка ↕, однако параметр команд дает возможность заменить ее другим символом. Первая команда размещает стрелку перед вертикалью выравнивания, а вторая за ней:

```
\begin{alignat*}{2}
   x+y&=a, \\ \ArrowBetweenLines
   x+z&=b, \\ \ArrowBetweenLines*
   y+z&=c.
\end{alignat*}
```

$$\begin{alignat*}{2} x+y&=a, \\ &\Updownarrow \\ x+z&=b, \\ &\qquad\Updownarrow \\ y+z&=c. \end{alignat*}$$

Еще одна конструкция связывает стоки формул многоточием «⋮». Ее основой являются команды, вставляющие многоточие, центрируя его по ширине символа, указанного в их аргументе:

\vdotswithin{символ} ,

\shortvdotswithin{символ} и \shortvdotswithin*{символ} ,

Для правильного центрирования символ должен быть идентичен символам, находящимся над и под многоточием:

```
\begin{align*}
  x &= a, \\
    & \vdotswithin{=} \\
  y &= x, \\
    \shortvdotswithin{=}
  z &= y.
\end{align*}
```

$$\begin{aligned} x &= a, \\ &\vdots \\ y &= x, \\ &\vdots \\ z &= y. \end{aligned}$$

Несколько команд `\vdotswithin` создают несколько связок:

```
\begin{equation}
  \begin{alignedat}{2}
    a_{11}x_1 & +\dotsb & a_{1n}x_n & = b_1,          \\[-1ex]
              & \vdotswithin+      && \vdotswithin=  \\[-1ex]
    a_{n1}x_1 & +\dotsb & a_{nn}x_n & = b_n.
  \end{alignedat}
\end{equation}
```

$$\begin{aligned} a_{11}x_1 &+ \dotsb & a_{1n}x_n &= b_1, \\ &\vdots && \vdots \\ a_{n1}x_n &+ \dotsb & a_{nn}x_n &= b_n. \end{aligned} \tag{9.15}$$

Команды `\vdotswithin` должны располагаться относительно вертикалей выравнивания, т. е. символов `&` так же, как и символы, которые они связывают. Вертикальная отбивка строк с многоточиями обычно выглядит избыточной, поэтому, как показано в примере, ее лучше уменьшить, используя параметр перевода строки с отрицательной длиной.

Команды `\shortvdotswithin` и `\shortvdotswithin*` вставляют одно многоточие, печатая его вблизи первой вертикали выравнивания. Они различаются лишь тем, что первая привязывает его к символу, стоящему справа от вертикали, а вторая — к символу, находящемуся слева. Данные команды автоматически подавляют нумерацию строки с многоточием и уменьшают расстояние между строками, сдвигая их на `2.15\jot`. Величину сдвига регулируют настройки

shortvdotsadjustabove = *длина*,
shortvdotsadjustbelow = *длина*.

9.8. Нумерация формул

Как видно на примере уравнения (9.6), длинное выражение вытесняет номер формулы в следующую строку. В многострочных конструкциях, если есть свободное место, его можно поднять вверх. Для этого пакет amsmath вводит команду `\raisetag{`*смещение*`}`, которую нуж-

но поставить в конце выражения. Положительная или отрицательная длина, указываемая в ее аргументе, смещает номер вверх или вниз.

Приведем пример коррекции положения номера формулы multline:

```
\begin{multline}
  x+y+z =  \\
    = a+b+ \\
  +c+d+e+f+g \raisetag{2.8\baselineskip}
\end{multline}
```

$$x+y+z = \quad (9.16)$$
$$= a+b+$$
$$+c+d+e+f+g$$

В окружении multline номер выводится в конце последней строки, но в примере она имеет большую длину, поэтому мы сместили его наверх, где он смотрится вполне органично.

Важно отметить, что команда `\raisetag` игнорируется, если номер умещается в стоке. Чтобы сдвинуть его в этом случае, поставьте в начале и конце формулы симметричные пробелы `\hspace*` и, подобрав их длину, вытесните номер из строки, тогда он окажется в вашем распоряжении.

Для коррекции *значений* номеров пакет amsmath определяет команды `\tag{`*текст*`}` и `\tag*{`*текст*`}`. Обе они печатают *текст* в поле, предназначенном для номера, но первая заключает его в скобки, а вторая — нет. Эти команды не обращаются к счетчику формул и не меняют его значение, они присваивают формуле «номер», указанный в их аргументе, например

```
\[ x=y tag{***}\label{e:*} \]
```

$$x = y \qquad (***)$$

Если рядом с ними поставить метку, ссылка на нее будет печатать этот номер: `\eqref{e:*}` ⇝ (***). Команды `\tag` действуют в любых вынесенных формулах. Устройство формулы в нашем примере не предусматривает нумерации, однако она «получила номер».

Скорректируем с помощью команды `\tag` номера связанных уравнений (4.3a) и (4.3b), генерируемые окружением subequations на с. 113. По умолчанию subequations помечает уравнения латинскими буквами, а мы заменим их арабскими цифрами:

```
\begin{subequations}\label{e:Tags}
   \begin{eqnarray}
      x &=& y+a
\tag{\ref{e:Tags}-1}\label{e:Tag1}\\
      z &=& a+x
\tag{\ref{e:Tags}-2}\label{e:Tag2}
   \end{eqnarray}
\end{subequations}
```

$$x = y + a \qquad (9.17\text{-}1)$$
$$z = a + x \qquad (9.17\text{-}2)$$

Вот как теперь выглядят ссылки на эти уравнения:

`\eqref{e:Tag1}` ⇝ (9.17-1), `\eqref{e:Tag2}` ⇝ (9.17-2).

Хотя команда `\tag` находится в формуле, ее аргумент является текстом, поэтому общий номер уравнений генерирует команда `\ref{e:Tags}`, ссылающаяся на номер, присвоенный окружению subequations. Индивидуальная часть номера добавлена вручную, но при необходимости этот процесс не сложно автоматизировать, введя и настроив дополнительный счетчик.

Пакет mathtools позволяет настроить формат вывода номеров. Для этого он вводит особый стиль и декларации, которые его создают, изменяют и применяют:

```
\newtagform{стиль}[шрифт]{левая скобка}{правая скобка} ,
\renewtagform{стиль}[шрифт]{левая скобка}{правая скобка} ,
\usetagform{стиль}.
```

Стиль default, применяющийся по умолчанию, печатает номера прямым шрифтом в круглых скобках. Новые стили создает декларация `\newtagform`, а `\renewtagform` служит для изменения уже существующих. Синтаксис обеих деклараций одинаков. В аргументах указываются имя стиля и скобки, в которые заключается номер. В параметре можно указать *команды* настройки шрифта для печати номера. Например, декларация

```
\newtagform{BracketsBold}[\textbf]{[}{]}
```

задаст вывод номера в квадратных скобках жирным шрифтом ⇝ [**5**]. Если вместо команд в параметре поставить *декларации*, они установят стиль печати не только номера, но и правой скобки. Если же декларации разместить в первом аргументе перед левой скобкой, они настроят печать номера и обеих скобок. Таким образом, декларация

```
\newtagform{BoldBrackets}{\bf[}{]}
```

задаст стиль печати номера в квадратных скобках жирным шрифтом.

Декларация `\usetagform` предназначена для активации созданного стиля, или возврата к стандартному:

`\usetagform{BoldBrackets}` ⇝ $x = y + a$ **[9.18]**

`\usetagform{default}` ⇝ $x = y + a$ (9.19)

Ее нужно использовать вне формул, а в математических окружениях она не работает, поэтому изменить стиль печати номера одного уравнения многострочной формулы с ее помощью не удастся.

Заданный стиль печати не применяется при выводе номера командами `\eqref` и `\ref`. Первая печатает его в скобках стандартным шрифтом документа, а вторая без скобок шрифтом текущего стиля текста. Чтобы напечатать номер стандартным шрифтом без скобок, пакет mathtools определяет команду `\refeq{`*метка*`}`.

Пакет mathtools дает возможность нумеровать только те формулы, на которые есть ссылки в документе. По умолчанию такой режим нумерации не активирован, его устанавливают настройки

showonlyrefs = true | **false**, showmanualtags = true | **false**.

Первая контролирует печать номеров, генерируемых автоматически, а вторая — проставляемых командами `\tag`. Режим showonlyrefs можно включать и выключать многократно. Он действует локально, его можно менять внутри группы, не опасаясь нарушить нумерацию вне ее. Разберем особенности его работы на следующем примере:

```
\mathtoolsset{showonlyrefs}
\begin{align}\label{e:xy}
  x &= y \\
  y &= z \\
  z &= x \tag{*}
  x &= a \label{e:xyz}
\end{align}
Равенство \eqref{e:xy} ... \noeqref{e:xyz}
\mathtoolsset{showonlyrefs = false}
\begin{equation}
  x=y+z
\end{equation}
```

$$x = y \quad (9.19)$$
$$y = z$$
$$z = a$$
$$x = a \quad (9.20)$$

Равенство (9.20) ...

$$x = y + z \quad (9.21)$$

В активированном вначале режиме showonlyrefs пронумерованы только первая и четвертая строка первой формулы. Номер четвертой обеспечила ссылка `\eqref{e:xyz}`, а первой — команда `\noeqref{e:xy}`. Команда `\noeqref{`*список меток*`}` позволяет нумеровать формулы без печати ссылок. В ее аргументе через запятую можно перечислить *список меток*. Настройка showonlyrefs действует и на номера, присваиваемые командой `\tag`, поэтому у третьей строки нет номера. Чтобы печать номеров `\tag` не зависела от режима нумерации, параметру showmanualtags нужно присвоить значение true. Перед второй формулой восстановлен обычный режим нумерации, поэтому она получила номер автоматически.

Ссылаясь на формулы в режиме showonlyrefs, не рекомендуется пользоваться командами `\ref`, иначе корректность его работы не гарантируется.

Завершая обзор дополнительных возможностей верстки формул, отметим, что конструкции пакета amsmath остаются неизменными уже несколько десятков лет и вряд ли изменятся в будущем. В то же время пакет mathtools развивается, его ресурсы постепенно наращиваются и интерфейс некоторых команд может со временем немного измениться. Если вы столкнетесь с тем, что поведение его конструкций отличается от их описания в данной главе, обратитесь к документации пакета [31].

Глава 10

Таблицы

В первой части книги верстка таблиц описана без лишних деталей, тем не менее представленные в ней методы и средства позволяют верстать таблицы, имеющие сложную структуру. В этой главе обсуждаются ресурсы, упрощающие набор таблиц и улучшающие их внешний вид. Мы рассмотрим методы регулирования ширины колонок и таблиц в целом, а также создание таблиц, которые могут занимать несколько страниц.

Прежде всего скажем, что для окружения tabular ядро LaTeX определяет не упоминавшуюся прежде конструкцию

`*{`*число*`}{`*последовательность колонок и разделителей*`}`,

упрощающую описание таблиц с большим количеством повторяющихся элементов. Их *последовательность*, указанная во втором аргументе, повторяется указанное *число* раз. Данная конструкция делает описание структуры таблицы более наглядным и компактным:

`\begin{tabular}{|lcr|lcr|lcr|lcr|}` ≡ `\{begin}{tabular}{ *4{|lcr} |}`.

Обратите внимание, что последнее отчеркивание, заданное разделителем `|`, исключено из повторений, чтобы между колонками не образовалась пара линий. Конструкции `*{•}{•}` можно вкладывать друг в друга, например конструкция `*2{ *2{|lcr} |}` эквивалентна последовательности `|lcr|lcr||lcr|lcr|`, вставляющей две отчеркивающие линии в середине таблицы. Неявно указанные аргументы наследуются аналогично командам LaTeX, поэтому конструкция `*8 l` задаст восемь колонок с левым выравниванием.

10.1. Расширение синтаксиса окружения tabular

Стандартный пакет array [72] из коллекции tools расширяет синтаксис аргумента окружения tabular и определяет ряд новых команд. Он вводит дополнительные типы колонок, еще один разделитель и операцию *вставки*. Перечисленные дополнения сведены в табл. 10.1.

Таблица 10.1
Дополнения к синтаксису окружения tabular

Спецификаторы колонок	Вставки
b{ширина} m{ширина}	>{код} — в начало ячеек <{код} — в конец ячеек
w{выравнивание}{ширина}	Разделитель
W{выравнивание}{ширина}	!{текст}

Новый разделитель !{•} аналогично разделителю @{•} вставляет текст между колонками, но в отличие от него сохраняет между ними стандартное пустое поле. Сравните, например:

```
\begin{tabular}{|r @{|} l|r !{|} l|} \hline
   1   & 2   & 3   & 4   \\
   10  & 20  & 30  & 40  \\
   100 & 200 & 300 & 400 \\ \hline
\end{tabular}
```

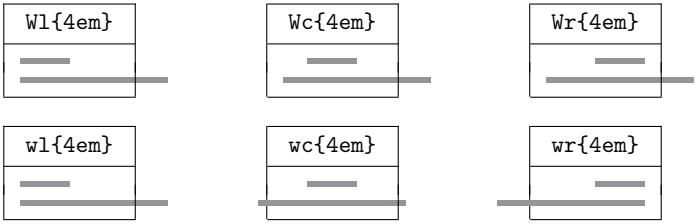
1|2	3 | 4
10|20	30 | 40
100|200	300 | 400

Вставки >{код} и <{код} позволяют выполнить ряд действий над всеми ячейками колонки. Первая из них должна находиться непосредственно перед спецификатором колонки, а вторая — после него. Код вставок добавляется, соответственно, в начало и конец ячеек.

Чаще всего вставки используются, чтобы изменить параметры верстки колонки. Например, конструкция >\tt r задаст печать текста колонки r шрифтом печатной машинки, а конструкция >$ l <$ заключит каждую ячейку колонки l в знаки доллара, поэтому она будет верстаться как формула. Отметим, что вставки >{•} и <{•} являются односимвольными командами, и, воспользовавшись правилом автоматического определения аргумента, можно опустить лишние скобки вокруг знаков доллара и команды \tt.

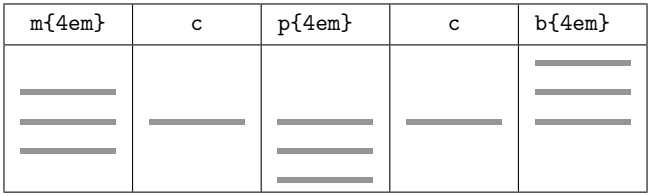
Пакет array вводит новые типы колонок фиксированной ширины w•{•} и W•{•}, которые содержат одну строку текста. Второй их ар-

гумент указывает ширину ячеек, а первый — тип выравнивания, задаваемый буквой: `l` — по левому краю, `r` — по правому краю, `c` — по центру. Колонки различаются своим поведением при переполнении. В колонках `W` вне зависимости от типа выравнивания длинная строка выступит за правый край, а в колонках `w` она будет выравниваться, как показано ниже:



Здесь короткие линии представляют строки, умещающиеся в ячейки, а длинные — выступающие за их границы.

Колонки `m{•}` и `b{•}` аналогичны стандартной колонке `p{•}` и отличаются от нее только методом встраивания в строку таблицы. Их ячейки могут содержать несколько строк текста, автоматически нарезаемого на строки заданной длины, выровненные справа и слева. Текст ячеек `m{•}` центрируется по высоте относительно табличной строки, а у ячеек `p{•}` и `b{•}` в нее встраивается, соответственно, первая и последняя строка текста. Это демонстрирует пример, в котором ячейки колонок `m, p` и `b` содержат по строки с длинами, равными ширине колонок, а ячейки колонок `c` — по одной:



Для печати численных данных стандартный пакет dcolumn [73] вводит колонку, в которой числа выравниваются по десятичному знаку:

`D{`*десятичный знак*`}{`*печатаемый десятичный знак*`}{`*шаблон*`}`.

Ее спецификатор имеет три аргумента, первым из которых является символ, разделяющий целую и дробную части числа в рукописи, а вторым — символ заменяющий его в документе. В третьем аргументе указывается число цифр, отводимое для печати целой и дробной частей,

например `5.5` или `2,5`. Колонки `D` верстаются в математической моде, поэтому их ячейки могут содержать любые математические символы.

Разберем особенности верстки таких колонок:

```
\begin{tabular}{|D,.{2.3} |D,.{2,1} |D,\cdot{-1}|} \hline
   0,125  & 0,125  & 0,125 \\
   10     & 10     & 10 \\
   29,768 & 29,768 & 29,768 \\ \hline
\end{tabular}
```

0.125	0.125	0·125
10	10	10
29.768	29.768	29·768

В первых двух колонках десятичная запятая заменяется точкой, а в последней — центрированной точкой. Шаблон первой колонки соответствует количеству дробной части чисел, а в шаблоне второй колонки для нее отведено недостаточно места, поэтому числа вылезли за правую границу колонки. Специальный шаблон `-1` колонок `D` предотвращает такое поведение. При его использовании компилятор определяет максимальное количество десятичных разрядов в дробных и целых частях всех чисел и отводит его для вывода обеих частей. В нашем примере оно равно трем, поэтому применение шаблона `-1` в третьей колонке привело к появлению пустого поля слева от чисел, целая часть которых содержит менее трех цифр. Приведенные примеры показывают, что, выравнивая числа по десятичному знаку, нужно внимательно следить за соответствием шаблона их формату.

Пакет array определяет команду, позволяющую вводить новые спецификаторы колонок:

`\newcolumntype{`*символ*`}[`*число аргументов*`]{`*описание*`}` .

Ее первый аргумент задает символ спецификатора, а параметр — число его аргументов. Описание может содержать любую комбинацию разделителей, вставок и уже известных идентификаторов. С помощью данной команды можно, например, упростить спецификатор выравнивания чисел с десятичной точкой: `\newcolumntype{d}[1]{D..{#1}}`. Тогда при использовании нового идентификатора достаточно будет указать только шаблон вывода: `d{2.3}`.

Пакет array помогает настроить правильное встраивание в текст таблиц в рамке. Как отмечалось в разд. 3.2.1, при использовании окружения `tabular` с параметром `b` или `t` в текстовую строку встраивается верх или низ рамки, а не табличная строка. Чтобы избежать этого, для верхней и нижней линии рамки введены команды `\firstline` и `\lastline`. Их использование дает ожидаемое поведение:

```
\begin{tabular}[•]{|c|c|} \firsthline
  a & c \\ \hline
  e & n \\ \lasthline
\end{tabular}
```

a	c
e	n

текст

a	c
e	n

.

10.2. Стандартные настройки

Перечислим команды, хранящие настройки окружений, формирующих таблицы. Большая часть из них является длинами. В скобках указаны значения параметров, установленные по умолчанию.

`\arraycolsep` (5pt) — полуширина пустого поля, разделяющего колонки в окружении `array`.

`\tabcolsep` (6pt) — полуширина пустого поля, разделяющего колонки в окружении `tabular` и всех его производных.

`\doublerulesep` (2pt) — ширина зазора между соседними отчеркивающими линиями `\hline`, `\vline` и «`|`».

`\arrayrulewidth` (0.4pt) — толщина отчеркивающих линий `\hline`, `\cline`, `\vline` и «`|`».

`\arraystretch` (1) — коэффициент, корректирующий межстрочный интервал в таблице.

`\extrarowheight` (0pt) — длина, добавляемая к высоте каждой строки таблицы. Вводится пакетом `array`.

Параметр `\arraystretch` хранит коэффициент, на который умножается дополнительный интервал, вставляемый сверху и снизу каждой строки таблицы. Его значение, по умолчанию равное единице, изменяется командой `\renewcommand\arraystretch{`*новое значение*`}`. Увеличивая или уменьшая его, можно раздвигать или сдвигать строки. Нулевое или отрицательное значение уберет дополнительный интервал, но при этом строки не наложатся друг на друга, а будут «соприкасаться».

10.3. Линии и рамки

10.3.1. Положение отчеркивающих линий

В большинстве случаев интервал между табличными строками настраивать не нужно, однако иногда слишком узкий зазор между

текстом и отчеркивающей линией хочется увеличить. К примеру, если ячейки содержат цифры или заглавные буквы, зазоры между ними и линиями, проведенными сверху и снизу строки, оказываются неодинаковыми:

```
\begin{tabular}{ccc} \hline
  MMM & KKK & ЖЖЖ \\ \hline
   33 & 55  & 77  \\ \hline
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

Рассмотрим методы, позволяющие управлять положением отчеркивающих линий. Глубину строки можно изменить с помощью параметра команды `\\[`*длина*`]`. Вставляемый дополнительный интервал сдвинет отчеркивающую линию вниз. Чтобы получить обратный эффект, т. е. сместить вниз строку относительно расположенной выше линии, нужно увеличить высоту строки распоркой. Для этого в одной из ее ячеек можно нарисовать фиктивную линию нулевой ширины, `\rule[`*глубина*`]{0pt}{`*высота*`}`, и подобрав ее *высоту* и *глубину*, добиться нужного смещения. Разделять распорку и тест ячейки пробелом не следует, так как он увеличит ширину ячейки.

Пакет booktabs [22], предлагая свой метод управления отбивкой линий, вводит команду `\addlinespace[`*длина*`]`, которая ставится в начале строки и смещает ее вниз. По умолчанию смещение составляет 0.5em, но параметр команды позволяет его регулировать. Команду `\addlinespace` можно использовать как после перевода строки командой `\\`, так и после команд, рисующих линии. Вот как с помощью `\addlinespace` можно добиться симметричного отчеркивания в таблице предыдущего примера:

```
\begin{tabular}{ccc}
\hline \addlinespace[.55ex]
  MMM & KKK & ЖЖЖ \\
\hline \addlinespace[.55ex]
   33 & 55  & 77  \\
\hline \addlinespace[.55ex]  \hline
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

Чтобы показать добавленную высоту, в конце таблицы прочерчены две линии. Команда `\addlinespace` настраивает зазор между отдельной парой линий, а глобально его устанавливает длина `\doublerulesep`.

Линии `\hline` и `\cline` пакет booktabs дополняет целым набором линий, которые не требуют коррекции и смотрятся много лучше. Как уже говорилось в разд. 3.2.3, он вводит команды `\toprule[`•`]`, `\bottomrule[`•`]`, `\midrule[`•`]` и `\cmidrule[`•`](`•`){`•`}`. Первая и

вторая команды рисуют более толстые линии и увеличивают их отбивку, соответственно, снизу и сверху. Остальные команды рисуют линии стандартной толщины, симметрично отбитые с обеих сторон. Вдобавок к перечисленным командам определена «универсальная линия», у которой настраивается как толщина, так и ширина обеих отбивок:

`\specialrule{`*толщина*`}{`*верхний зазор*`}{`*нижний зазор*`}`.

Покажем наглядно преимущества линий пакета booktabs, использовав их в нашем примере:

```
\begin{tabular}{ccc} \toprule
  MMM & KKK & ЖЖЖ \\ \midrule
   33 & 55  & 77  \\ \bottomrule
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

В ядре LaTeX отсутствует возможность многократного отчеркивания диапазона ячеек. Пакет booktabs устраняет эту недостачу, определяя команду `\morecmidrules` и новую длину `\cmidrulesep`, по умолчанию равную `\doublerulesep`. Чтобы дважды отчеркнуть ячейки, команду `\morecmidrules` нужно поставить между командами `\cmidrule`:

```
\begin{tabular}{ccc}  \toprule
  MMM & KKK & ЖЖЖ \\
    \cmidrule{1-1} \cmidrule{3-3}
    \morecmidrules \cmidrule{3-3}
   33 & 55  & 77  \\  \bottomrule
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

Разделитель `@{•}` позволяет отчеркивать столбцы вертикальными линиями, регулируя ширину пустых полей вокруг них. Для этого в его аргумент нужно поместить команды `\vline` (см. далее), окружив их пробелами нужной ширины, например `@{\hspace{0.5em} \vline \quad}`.

10.3.2. Толщина линий

Толщина линий в таблицах настраивается довольно просто. Изменяя длину `\arrayrulewidth`, можно глобально управлять толщиной линий `|`, `\hline`, `\cline`, однако задать толщину отдельной линии таким образом не удастся, так как внутри таблицы изменить значение `\arrayrulewidth` невозможно. С этой целью у команд `\bottomrule[•]`, `\toprule[•]`, `\midrule[•]` и `\cmidrule[•]` предусмотрен параметр, позволяющий регулировать толщину каждой линии (см. с. 72).

Регулировать толщину вертикальных линий позволяет имеющая нестандартный синтаксис команда `\vline width` *толщина*, опреде-

ленная в ядре LaTeX. Ее параметр `width` устанавливает *толщину* рисуемой линии, а если он опущен, используется значение `\arrayrulewidth`. В разделителях `@{•}` и `!{•}` она рисует линию по всей высоте таблицы, в ячейках — на глубину и высоту ячейки, а в обычном тексте — на глубину и высоту строки.

10.3.3. Пересечение линий

Предваряя обсуждение верстки рамок, еще раз обратим внимание на то, что вертикальные линии загромождают таблицы и мешают восприятию данных, поэтому в своих требованиях к оформлению таблиц большинство издательств рекомендует их избегать. Вследствие этого пакет booktabs не расширяет возможности вертикального отчеркивания, более того, введенные им команды разбивают вертикальные линии.

Средства, имеющиеся в ядре LaTeX, давая возможность разлиновать таблицу одиночными, двойными, тройными и т. д. линиями, не позволяют реализовать разные варианты их пересечений. С этой целью стандартный пакет hhline [74] вводит команду `\hhline{`*описание линии*`}`, которая не только рисует горизонтальные линии, но и настраивает их сочленение с вертикальными. Для этого линия разбивается на сегменты, которые в простейшем случае представляют собой отрезки, отчеркивающие ячейки, описываемые своеобразной мнемоникой:

`-` обычная линия, `=` двойная линия, `~` отсутствие линии.

В аргументе команды `\hhline` перечисляются сегменты, отчеркивающие ячейки всех колонок:

```
\begin{tabular}{|c|c|c|} \hline
  MMM & KKK & ЖЖЖ \\ \hhline{-~=}
   33 & 55  & 77  \\ \hline
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

В данном примере таблица отчеркивается вверху и внизу обычными линиями `\hline`, а ее строки разделяет команда `\hhline`, формирующая одиночную линию в первом сегменте и двойную в последнем. Средняя ячейка не отчеркивается. Сопряжение горизонтальных и вертикальных линий здесь не задано, поэтому между линиями возникли зазоры.

Последовательность одинаковых сегментов можно «свернуть», воспользовавшись конструкцией `*{`*число*`}{`*описание сегментов*`}`, повторяющей второй аргумент указанное *число* раз. Например, команду `\hhline{=~-=~-=~-=~-=~-}` можно сократить до `\hhline{*5{=~-}}`.

Когда команда `\hhline` прочерчивает линию, состоящую из одиночных линий и пустых сегментов, ее сочленение с вертикальными линиями определено однозначно и потому специального описания не требует. Пересечение пары двойных линий образует двойной крест, сердцевиной которого является квадрат, состоящий из четырех отрезков-сочленений, каждый из которых можно прочертить или опустить. Такое пересечение реализуется разными способами, и чтобы задать его однозначно, нужно описать отрезки, соединяющие линии. В описаниях сочленений используется своя мнемоника:

`t` — рисовать верхнюю линию, `#` — рисовать все линии,
`b` — рисовать нижнюю линию, `:` — сочленить линии точками,
`|` — рисовать левую или правую линию.

Варианты сочленений представлены на рис. 10.1 в верхнем ряду. В центре каждого примера приведены описания операций, обеспечивающих показанные комбинации линий. Слева показан пустой «квадрат пересечений» двойного креста. Вертикальные и горизонтальные линии, отчеркивающие ячейки, соприкасаются без пересечения, а чтобы сочленить их между собой используется двоеточие. Далее следуют варианты прорисовки сторон квадрата.


```
\begin{tabular}{||c||} \hhline{...}
  \cellcolor{Bckgd} \\ \hhline{...}
\end{tabular}&
```

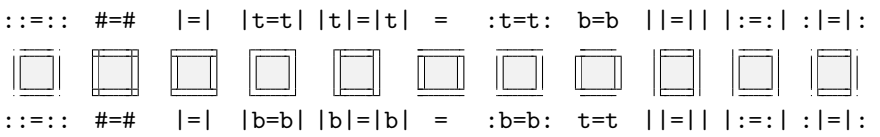


Рис. 10.1. Варианты пересечения линий `\hhline`

Логика формирования сочленений довольно проста: между сегментами, задающими отчеркивание ячеек, описываются стороны «квадрата пересечений», а если описания отсутствуют, рисуются только горизонтальные сочленения. Используя перечисленные средства, мы составили разные варианты обрамления ячеек. Примеры, расположенные внизу рис. 10.1, оформлены в виде «таблиц», состоящих из одной ячейки,

залитой фоном (см. разд. 10.6). Сверху и снизу приведены описания, содержащиеся в аргументах первой и второй команд \hhline.

В крайней левой позиции показано сочленение линий без пересечения. В таблицах, состоящих из нескольких колонок и строк, точки, стоящие в «пустых углах», сливаются с линиями, обрамляющими соседние ячейки. Четыре следующие позиции занимают варианты сочленения вертикальных и горизонтальных линий, за ними следуют три позиции с горизонтальными сочленениями, а в последних трех позициях представлены только вертикальные сочленения. Отметим, что пересечение линий, показанное в последнем примере, без точек, порождаемых двоеточиями, не реализуется.

Применим команду \hhline и распорку \bigstrut в заключительном варианте оформления нашей таблицы:

```
\begin{tabular}{||c||c||c||} \hhline{|t:=:t:=:t:=:t|}
  MMM & KKK & ЖЖЖ \bigstrut[t] \\
      \hhline{||-||~|:=:|}
   33 & 55  & 77  \bigstrut[t] \\
      \hhline{|*3{b:=:}b|}
\end{tabular}
```

MMM	KKK	ЖЖЖ
33	55	77

10.4. Длинные таблицы

Как говорилось ранее, таблицу, созданную окружением tabular, нельзя разбить на части. Чтобы снять это ограничение, стандартный пакет longtable [75] определяет окружение longtable:

\begin{longtable}[выравнивание]{шаблон строки},

верстающее таблицу, которая может занимать несколько страниц. Еще одним важным дополнением является возможность сформировать подпись таблицы непосредственно в окружении longtable. Сверстанной таблице присваивается номер. Она не является плавающим объектом, ее положение в документе соответствует положению в рукописи. Все описанные ранее методы верстки применимы и для длинных таблиц. Синтаксис описания колонок и разделителей в окружениях tabular и longtable одинаков. В простейшем случае можно просто заменить окружение tabular на longtable, однако последнее имеет гораздо больше возможностей. Примеры длинных таблиц содержат прил. Б и В.

Параметр окружения longtable, задающий метод выравнивания таблицы в тексте, может принимать следующие значения:

`c` — таблица центрируется в колонке текста (по умолчанию);
`l` — таблица прижимается к левому краю;
`r` — таблица прижимается к правому краю.

Изменение длин `\LTpre`, `\LTpost`, `\LTleft` и `\LTright` позволяет настроить отбивку длинных таблиц по своему усмотрению. Пробелы с длинами `\LTpre` и `\LTpost`, по умолчанию равными `\bigskipamount`, ставятся сверху и снизу таблиц. Пробелы с длинами `\LTleft` и `\LTright` окружают их по бокам. Этим длинам присваивается бесконечно растяжимая длина `\fill`, превращающая пробелы в пружины, центрирующие таблицы на странице или в колонке текста. Параметры `l` и `r` устанавливают одну из длин в нуль, при этом вторая не меняется и оставшаяся пружина прижимает таблицу налево или направо.

Таблица может иметь подпись, оформляемую командами

`\caption[`*краткая подпись*`]{`*подпись*`}` и `\caption*{`*подпись*`}`,

Так же как и в окружении table, команда `\caption` ассоциирована со счетчиком таблиц. В ее параметре можно указать сокращенный вариант подписи для списка таблиц. Чтобы сослаться на номер таблицы, поставьте после нее метку `\label{•}`. При использовании команды `\caption*` таблица не нумеруется и в список таблиц не включается. В окружении longtable после подписи (и метки) нужно поставить команду перевода строки `\\`. Длина `\LTcapwidth` регулирует ширину подписей. По умолчанию ее значение равно четырем дюймам.

Рассмотрим организацию длинной таблицы (табл. 10.2). Помимо подписи у нее может быть главный заголовок, выводимый только в самом начале. Каждая новая часть должна начинаться пояснением, является ли она продолжением, или окончанием таблицы, а также описанием названий колонок. Такими образом, начало, продолжение и окончание таблицы оформляются по-разному, ее конец и разрывы на промежуточных страницах также имеют разный вид.

Таблица 10.2

Пример организации длинной таблицы[1]

Главный заголовок[2]		
Колонка 1	Колонка 2	Колонка 3
Ячейки	первой	строки

[1] Пример сноски в подписи таблицы.
[2] Пример сноски в заголовке таблицы.

Окончание табл. 10.2

Колонка 1	Колонка 2	Колонка 3
Ячейки	второй	строки[3]
Ячейки	последующих	строк…

Верстая длинную таблицу, компилятор сам определит места ее разбиения, если они не указаны явно командами `\newpage` или `\pagebreak[•]`. Их ставят между строками таблицы, как показано далее в коде примера. Ширина колонок, расположенных на разных страницах, выравнивается за несколько проходов компиляции. Сначала таблица разбивается на несколько частей, формируемых независимо друг от друга, затем для каждой колонки определяется максимальное значение ширины, вычисленной в разных частях, и на конечном этапе ширина более узких колонок увеличивается до более широких. По умолчанию части состоят из 200 строк, количество которых при необходимости можно увеличить, чтобы ускорить процесс формирования таблиц. Для этого нужно изменить значение счетчика `LTchunksize`, хранящего число строк, к примеру `\setcounter{LTchunksize}{500}`.

В длинной таблице можно сформировать преамбулу, содержащую подпись и элементы с описаниями того, как оформляются начало и конец таблицы, а также ее заголовки и разрывы на промежуточных страницах. Команды `\endfirsthead`, `\endlastfoot`, `\endhead` и `\endfoot` отделяют элементы друг от друга, а сами они состоят из обычных табличных строк. Порядок следования элементов в преамбуле не важен, компилятор автоматически расставляет их по назначению. Ни один из элементов не обязателен, любой можно опустить.

Преамбулу табл. 10.2 составляют подпись и главный заголовок, выведенные только в ее начале, а также комментарий «Окончание табл…» и заголовки колонок, напечатанные на каждой странице. Перед обсуждением кода, сверставшего данную таблицу, отметим особенности формирования сносок в длинных таблицах. В ячейках для их печати используются команды `\footnote`. В преамбуле сноски нужно разбить на метки `\footnotemark` и текст, размещаемый в командах `\footnotetext`, которые должны находиться в одной из ячеек таблицы *за* преамбулой, но *до* сносок в ячейках.[4] Последнее требование необходимо для правильной нумерации сносок. Сноски выводятся внизу страниц, на которых

[3] Пример сноски в ячейке.

[4] Команды `\footnotetext` лучше всего разметить в первой ячейке за преамбулой.

находятся ячейки, содержащие команды \footnotetext и \footnote. Команды \footnotemark и \footnotetext успешно справляются с задачей, когда в преамбуле находится одна сноска, но если их больше, как в нашем примере, для их нумерации понадобится дополнительный счетчик:

```
\newcounter{tablenote}
\begin{longtable}{ccc}
%----------------------- преамбула ---------------------
\caption[Пример организации длинной таблицы]%
{Пример организации длинной таблицы\footnotemark%
\setcounter{tablenote}\thefootnote}
\label{t:LongTable}
   \\ \toprule
      \multicolumn{3}{c}{Главный заголовок\footnotemark}
   \\ \midrule
      Колонка 1 & Колонка 2 & Колонка 3
   \\ \midrule
\endfirsthead
      \multicolumn{3}{r}{Окончание табл. \thetable}
   \\ \midrule
      Колонка 1 & Колонка 2 & Колонка 3
   \\ \midrule
\endhead
   \midrule
\endfoot
   \bottomrule
\endlastfoot
%--------------------- табличные данные -------------------
\footnotetext[\thetablenote]{Пример сноски в подписи таблицы.}%
\refstepcounter{tablenote}%
\footnotetext[\thetablenote]{Пример сноски в заголовке таблицы.}%
   Ячейки &   первой    & строки \\
\pagebreak
   Ячейки & второй      & строки\footnote{Пример сноски в ячейке.}\\
   Ячейки & последующих & строк...
\end{longtable}
```

Таблица содержит три сноски, две из которых находятся в преамбуле. Для их подсчета вводится новый счетчик tablenote. Одна из сносок находится в подписи таблицы, поэтому команда \caption имеет параметр, в который помещается подпись без номера сноски. В ее аргументе

команда `\footnotemark` обновляет и печатает номер сноски, а команда `\setcounter` присваивает его счетчику tablenote.

За подписью следует описание главного заголовка, состоящее из отчеркивающей линии `\toprule`, табличной строки с «Главным заголовком», помещенным в центрированную ячейку с тремя объединенными колонками, и строки с наименованиями колонок, отчеркнутой сверху и снизу линиями `\midrule`. Завершает описание команда `\endfirsthead`.

Описание заголовка, открывающего продолжение таблицы, содержит строку с объединенной ячейкой с правым выравниванием, в которую помещен заголовок «Окончание табл.» и команда `\thetable`, выводящая номер таблицы. Затем следует строка с названиями колонок, отчеркнутая сверху и снизу. Завершает описание команда `\endhead`.

При разрыве таблицы рисуется отчеркивающая линия `\midrule`, и выполняется команда `\endfoot`.

Последнюю строку таблицы отчеркивает команда `\bottomrule`, стоящая в конце преамбулы, завершаемой командой `\endlastfoot`.

Команда `\pagebreak` разрывает таблицу и инициирует вывод ее заголовка на следующей странице. Она поставлена для наглядности, так как компилятор автоматически разбил бы таблицу в этом месте.

Первая ячейка таблицы содержит команды `\footnotetext`, выводящие текст сносок преамбулы. Стоящая в их параметре команда `\thetablenote` печатает сохраненное значение счетчика сносок, при этом перед выводом второй сноски команда `\stepcounter` увеличивает его на единицу. Столь сложная организация нумерации сносок позволяет синхронизировать их с остальными сносками документа, но как говорилось ранее, если преамбула таблицы содержит лишь одну сноску, никаких дополнительных мер синхронизации ее сносок с нумерацией сносок документа не требуется.

10.5. Таблицы фиксированной ширины

Ширина обсуждавшихся ранее таблиц изменяется в зависимости от содержимого их ячеек. При необходимости ее можно зафиксировать, используя в таблице только колонки типа `p{•}` или `w{•}{•}`.

В простейшем случае таблицу фиксированной ширины можно составить из колонок `w{•}{•}` с разным типом выравнивания, но нужно внимательно следить за их возможным переполнением.

Колонки `b{•}`, `m{•}` и `p{•}` позволяют сверстать таблицу с ячейками, содержащими несколько строк. С помощью левой вставки с соответству-

ющей декларацией можно установить требуемое выравнивание колонок, например центрирование: `>\centering p{•}`. Текст ячеек будет автоматически нарезаться на строки и выравниваться, а для разрыва строк можно пользоваться командами \\, `\newline` или `\linebreak`. В этом случае, чтобы не запутаться, строки таблицы лучше заканчивать командой `\tabularnewline[длина]`, специально предназначенной для этих целей. Ее параметр позволяет регулировать расстояние между строками, аналогично команде `\\[•]`. Покажем на простом примере, как сделать подобную таблицу:

```
\begin{tabular}{|>\centering m{5em} |wl{5em}| >\raggedleft m{5em}|}
\hline
   колонка 1     & 1-я строка & колонка 2 \tabularnewline
\hline
   колонка \\ 1  & 2-я строка & колонка 3 2-я строка \tabularnewline
\hline
\end{tabular}
```

колонка 1	1-я строка	колонка 3
колонка 1	2-я строка	колонка 3 2-я строка

Декларации `\centering` и `\raggedleft` обеспечивают центрирование и правое выравнивание первой и последней колонок. Средняя автоматически выравнивается по левому краю. В первой колонке вторая строка разорвана командой \\, тогда как в третьей компилятор перевел ее автоматически.

Для верстки таблиц фиксированной ширины предназначено окружение `tabular*`:

`\begin{tabular*}{`*ширина*`}[`*привязка*`]{`*описание*`}`,

определенное в ядре LaTeX. Ширина указывается в его первом аргументе, а в остальном синтаксис и правила верстки `tabular*` полностью идентичны `tabular`. Окружение `tabular*` отводит под таблицу объявленную *ширину* и позволяет расширить таблицу, если она окажется уже, чем нужно. Если же ширина превысит объявленную, таблица вылезет за пределы отведенного ей места и наложится на окружающий текст. Нужный размер подгоняется за счет увеличения пустого поля между колонками. Для этого предусмотрен дополнительный пробел `\extracolsep{`*длина*`}`, вставляемый слева от колонок. По умолчанию его длина равна нулю, но если заполнить `\extracolsep` бесконечно растяжимой длиной `\fill`, пробелы преобразуются в пружины и расстояние между колонками автоматически подстраивается под нужный

размер. Чтобы изменить величину пробела, нужно использовать разделители `@{•}` и `!{•}`. Поясним сказанное несколькими примерами:

```
\begin{tabular*}{12em}{|c|c|c|} \hline
   1  &  2  &  3  \\
  111 & 222 & 333 \\ \hline
\end{tabular*}
```

1	2	3	
111	222	333	

```
\begin{tabular*}{12em}{@{\extracolsep\fill}|c|c|c|}\hline
   1  &  2  &  3  \\
  111 & 222 & 333 \\ \hline
\end{tabular*}
```

1	2	3
111	222	333

```
\begin{tabular*}{12em}{@{\extracolsep\fill}|ccc|} \hline
   1  &  2  &  3  \\
  111 & 222 & 333 \\ \hline
\end{tabular*}
```

1	2	3
111	222	333

```
\begin{tabular*}{12em}{|c|c|!{\extracolsep\fill}c|}\hline
   1  &  2  &  3  \\
  111 & 222 & 333 \\ \hline
\end{tabular*}
```

1	2	3
111	222	333

```
\begin{tabularx}{12em}{|X|X|X|} \hline
   1  &  2  &  3  \\
  111 & 222 & 333 \\ \hline
\end{tabularx}
```

1	2	3
111	222	333

В первом случае пробел `\extracolsep` имеет нулевую ширину, и пустое поле между колонками не увеличивается. В то же время, ширина таблицы соответствует заданному размеру 12pt, о чем свидетельствует длина линий, отчеркивающих ее сверху и снизу. Во втором примере пробел `\extracolsep` преобразуется в пружины, и колонки раздвигаются. Видно, что перед первой колонкой пружина отсутствует, а перед другими колонками они стоят слева, и вертикальные отчеркивающие линии выглядят несимметрично, тогда как в отсутствие отчеркивающих линий отбивки колонок смотрятся сбалансировано. Это показывает третий пример. Предпоследний пример демонстрирует, что команда `\extracolsep` является декларацией, поэтому ширина полей между колонками, находящимися до нее, т. е. до разделителя, в котором устанавливается ее новое значение, не меняется.

В последнем примере использовано окружение tabularx:

\begin{tabularx}{*ширина*}[*привязка*]{*описание*},

которое вводит одноименный пакет tabularx. В нем определены колонки X, ширина которых подбирается так, чтобы обеспечить нужный размер таблицы. Колонки X суть колонки p{\hsize}, ширина которых подгоняется за несколько итераций. Команда \hsize, наследованная из TEX, хранит текущую длину, установленную для верстки текста: длину строки, ширину бокса или ячейки и т. д.

Так как метод создания таблиц, реализованный в окружении tabularx, ни в чем не превосходит и даже уступает способам, описанным в начале данного раздела, мы не будем обсуждать его более подобно и отошлем заинтересованного читателя к книгам [3, 4] и документации пакета [76].

10.6. Цвет в таблицах

Цветную верстку таблиц обеспечивает пакет colortbl [77]. Для этого он вводит ряд команд, раскрашивающих колонки, строки и отдельные ячейки. Цвета отдельных элементов оформления, как то фон, текст и линии, устанавливаются самостоятельно.

Декларация \columncolor[*палитра*]{*цвет*}[*слева*][*справа*] задает фон колонки. По умолчанию колонка заливается фоном целиком, включая ячейки и окружающие их поля. Параметры регулируют ширину заливки ее полей. Для симметричной заливки достаточно указать один параметр. Отрицательные длины, заданные в параметре, уменьшают ширину заливки ячеек. Чтобы задать фон колонки, декларацию \columncolor нужно поместить во вставку >{•}. Рассмотрим особенности раскрашивания колонок на следующем примере:

```
\mbox{}\hfill \begin{tabular}{| >{\columncolor{Bckgd}} c
       | >{\columncolor{Bckgd}[0pt]} c
       | >{\columncolor{Bckgd}[0pt][\tabcolsep]} c
       | c | >{\columncolor{Bckgd}[2\tabcolsep]} c |}
\hline
   000 & 111 & 222 & 333 & 444 \\
   555 & 666 & 777 & 888 & 999 \\
\hline
\end{tabular}
```

000	111	222	333	444
555	666	777	888	999

Фон первой колонки, заданный по умолчанию, заливает ячейки и поля. У полей второй колонки фон отсутствует, так как в парамет-

ре команды `\columncolor` указана нулевая длина. В третьей колонке установлен фон ячейки и правого поля. Последняя пара колонок демонстрирует, что параметры таблицы не ограничивают ширину заливки. По умолчанию ширина поля, разделяющего колонки, равна удвоенной длине `\tabcolsep`, а поле, отбивающее таблицу (т. е. крайние колонки) от окружающего текста, имеет ширину `\tabcolsep`. В нашем примере ширина цветной заливки последней колонки слева и справа составляет `2\tabcolsep`, поэтому фон занимает все пространство между ней и предпоследней колонкой, а так как поле, отделяющее ее ячейки от правой отчеркивающей линии, оказывается вдвое уже, фон вылезает за правый край таблицы.

Порядок выполнения операций влияет на раскраску. При формировании табличной строки операции выполняются слева направо, как указано в аргументе окружения `tabular`: сначала печатается самый левый разделитель, затем колонки и их разделители, а в конце выводится самый правый разделитель. При раскраске *каждой* колонки сначала выводится фон, а на него накладывается текст ячейки и правый разделитель. В соответствии с этим алгоритмом линию, разделяющую в нашей таблице две последние колонки (см. `c | >{...` в строке перед первой командой `\hline`), закрывает фон правой колонки, а линия, отчеркивающая таблицу, рисуется поверх фона. При большой ширине заливки фон правой колонки может закрыть и часть ячеек левой колонки. Линии `\hline` и `\hhline` рисуются поверх фона, а вот линии `\cline` он закрывает. В цветных таблицах вместо них лучше пользоваться линиями `\hline{...-...}`. Раскраска не влияет на размеры, таблицы, поэтому фон, выступающий за ее границы, может наложиться на окружающий текст. В нашем примере таблица отжата направо жесткой пружиной, правая отчеркивающая линия совпадает с границей страницы, а фон выступает за нее.

Декларации `\rowcolor[`*палитра*`]{`*цвет*`}[`*слева*`][`*справа*`]` задают фон строк. По умолчанию фон заливает строку в целом, но если заданы параметры, ячейки заливаются целиком, а их поля на указанную ширину. Установки параметров одинаково действуют на все ячейки. При отрицательных значениях длин, заданных в параметрах, границы заливки ячеек сдвигаются внутрь. Как и в случае `\columncolor`, для симметричных полей достаточно указать одну длину.

Декларации `\rowcolor` раскрашивают отдельные строки, их нужно ставить в начале каждой выделяемой строки. Для раскраски последовательности строк определены декларации

`\rowcolors[`*команды*`]{`*номер*`}{`*цвет нечетн. стр.*`}{`*цвет четн. стр.*`}`,
`\rowcolors*[`*команды*`]{`*номер*`}{`*цвет нечетн. стр.*`}{`*цвет четн. стр.*`}`.

В их первом аргументе указывается номер строки, с которой начинается раскраска. Второй и третий аргументы задают цвет фона нечетных и четных строк. Если цвет не указан, фон отсутствует. В параметр можно поместить команды `\hline`, которые будут выполняться при выводе строк. Декларации `\rowcolors` сделают это для каждой строки, а `\rowcolors*` только для раскрашенных. Используя в начале строк декларации `\showrowcolors` и `\hiderowcolors`, раскраску можно прерывать и возобновлять:

```
\rowcolors*[\hline]{2}{MyCyan}{LightYellow}
\begin{tabular}{l}
   строка \therownum \\
   строка \therownum \\
   строка \therownum \\ \hiderowcolors
   строка \therownum \\
   строка \therownum \\ \showrowcolors
   строка \therownum \\
   строка \therownum
\end{tabular}
```

строка 1
строка 2
строка 3
строка 4
строка 5
строка 6
строка 7

Номера строк в примере подсчитывает счетчик rownum, введенный пакетом colortbl. Его значения изменяются только в области действия деклараций `\rowcolors` и `\rowcolors*`, причем каждая новая декларация сбрасывает их.

Декларации `\rowcolors` и `\rowcolors*` должны находиться в тексте *перед* таблицей. Если не ограничить область их действия, они раскрасят не только саму таблицу, но и все, следующие за ней. Использовав декларации с пустыми аргументами, можно отменить раскраску, при этом установленное отчеркивание строк и их подсчет счетчиком rownum останутся в силе. При совместном действии деклараций `\rowcolors` и `\rowcolor` приоритет имеют последние.

Фон ячейки задает декларация `\cellcolor[`*палитра*`]{`*цвет*`}`. Ячейка раскрашивается вместе с полями, ширина которых не регулируется. Когда разные варианты раскраски налагаются друг на друга, фон строки перекрывает фон колонки, а фон ячейки перекрывает оба. Это иллюстрирует еще одна таблица, в ячейках которой указаны декларации, формирующие фон. Первая ячейка первой строки фона не имеет, но фоном выделен ее текст, помещенный в цветной бокс `\colorbox`. В этом случае у полей, разделяющих колонки, фона нет. Фон второй

ячейки устанавливает декларация \columncolor. Со второй строки начинают действовать установки декларации \rowcolors, однако в самой второй строке их сначала перекрывает декларация \rowcolor, задающая фон первой ячейки, а затем декларация \cellcolor «отменяет» раскраску второй ячейки, заливая ее белым фоном. Самостоятельная операция аннулирования фона не предусмотрена. Фон двух последних строк задает декларация \rowcolors:

```
\begingroup \arrayrulewidth 1pt
\color{Sepia}
\rowcolors[\hline]{3}{MyCyan}{LightYellow}
\begin{tabular}{l | >{\color{blue}\columncolor{Bckgd}} l}
   \cmd{colorbox}  & \cmd{columncolor} \\
\rowcolor{Bckgd}
   \cmd{rowcolor}  &
\cellcolor{white}\color{black}
                     \cmd{cellcolor}   \\
   \cmd{rowcolors} & \cmd{rowcolors}   \\
   \cmd{rowcolors} & \cmd{rowcolors}
\end{tabular}
\endgroup
```

\colorbox	\columncolor
\rowcolor	\cellcolor
\rowcolors	\rowcolors
\rowcolors	\rowcolors

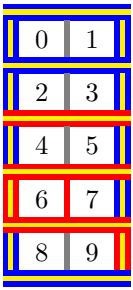
Имена команд здесь напечатала введенная на с. 158 команда \cmd.

Наш пример также иллюстрирует настройку цвета текста. Используя декларацию \color, его можно задать для колонок, отельных ячеек и таблицы в целом. К сожалению, настроить цвет шрифта в строках, привлекая только стандартные средства, не получится. В колонках цвет текста устанавливают с помощью вставок: >{\color{...}}. Аналогично можно задать его вместе с фоном: >{\columncolor{...}\color{...}}. В ячейках декларация \color ставится перед текстом, а чтобы изменить цвет шрифта всей таблицы, поставьте декларацию перед ней и не забудьте ограничить область ее действия. При «вложении» цветов главный приоритет имеет установка, сделанная в ячейке, а самый низкий — цвет, заданный для таблицы. Это тоже показано в примере, где для текста таблицы установлен коричневый цвет, для второй колонки задан синий цвет, а во второй ячейке второй строки — черный.

По умолчанию отчеркивающие линии раскрашиваются цветом, заданным для текста таблицы. Цвет отдельных вертикальных линий можно изменить, воспользовавшись разделителем !{...} или @{...}, декларацией \color и командой \vline, например !{\color{red}\vline} для красной линии. Пакет colortbl позволяет настроить цвета всех линий. Для этого он вводит декларации \arrayrulecolor{*палитра*}[*цвет*]

и `\doublerulesepcolor{`*палитра*`}[`*цвет*`]`, устанавливающие цвет горизонтальных и вертикальных линий и промежутков между ними. Их можно использовать как вне таблиц, так и внутри них. Они являются глобальными, поэтому, изменив цвета, вернуться к прежним настройкам можно, лишь задав их заново. Обсудим раскраску линий на примере, иллюстрирующем эту операцию:

```
\arrayrulecolor{blue}\doublerulesepcolor{yellow}
\begingroup \arrayrulewidth 2pt
\begin{tabular}{ || c !{\color{gray}\vline} c || }
              \hline \hline
   0 & 1   \\ \hline \hline
   2 & 3   \\ \arrayrulecolor{red} \hhline{==}
   4 & 5   \\ \hline \hline
   \multicolumn{1}{||c|}{6} & 7 \\ \hhline{==}
   8 & \multicolumn{1}{c||}{9}
          \\ \hhline{ >{\arrayrulecolor{blue}} == }
\end{tabular}
\endgroup
\doublerulesepcolor{white}\arrayrulecolor{black}
```



Для наглядности толщина линий увеличена до ширины зазора между ними, равного 2pt, поэтому вместе с раскрашенным зазором две линии смотрятся как три. Область изменения толщины ограничена группой, а для установки ее значения декларацией `arrayrulewidth 2pt` использованы краткие нотации TEX. Линии, разделяющие строки, прочерчены командами `\hline` и `\hhline`, чтобы показать, что их можно использовать вместе друг с другом. Линии по бокам таблицы заданы разделителями `|`, а центральная — разделителем `!{\color{gray}\vline}`, рисующим серую линию.

Декларации `\arrayrulecolor` и `\doublerulesepcolor` в начале примера устанавливают синий цвет линий и желтый цвет промежутка между ними. Так отчеркиваются верх таблицы и ее первая строка. Затем декларация `\arrayrulecolor` меняет цвет линий на красный для отчеркивания второй и последующих строк. При отчеркивании последней строки цвет вновь меняет вставка `>{•}`, которую пакет colortbl добавляет к синтаксису команды `\hhline{...}`. Разместив в ней обсуждаемые декларации, можно придать индивидуальный цвет любому сегменту линии, например

```
\hhline{ >{\arrayrulecolor{•}} |t= >{\arrayrulecolor{•}} :t:= ...}.
```

В нашем случае цвет изменяется перед первым сегментом, а так как декларация `\arrayrulecolor` является глобальной, то он устанавлива-

ется не только для всей линии, но и для всех последующих. В конце примера восстанавливается стандартный черный цвет линий и белый цвет промежутка.

Изменение цвета линий *внутри* таблицы не затрагивает вертикальные линии. Как видно на примере третьей строки, декларация `\arrayrulecolor{red}` не изменила цвет ни одной из них. Чтобы настроить цвет вертикальных линий внутри таблицы, нужно использовать команду `\multicolumn`. С ее помощью задан цвет линий ячеек 6 и 9. Ячейка 9 показывает, что после изменений, внесенных командой `\multicolumn`, цвета линий восстанавливаются к первоначальным.

В предыдущем примере показано несколько вариантов настройки цвета линий таблиц. На практике он, как правило, устанавливается для таблицы в целом, и иногда цветной рамкой выделяются отдельные ячейки. Описанные средства пакетов colortbl и hhline позволяют это сделать:

```
\begingroup
\arrayrulewidth 1pt                \def\RedVline{\color{red}\vline}
\def\Red{\arrayrulecolor{red}}     \def\Black{\arrayrulecolor{black}}
\begin{tabular}[b]{|c|c|c|} \hline
   1 & 2 & 3 \\ \cline{1-1} \Red \cline{2-2} \Black \cline{3-3}
   \multicolumn{1}{|c !{\RedVline}}{4} &
   \multicolumn{1}{ c !{\RedVline}}{5} &
           6 \\ \hhline{ - >{\Red} - >{\Black} - }
   7 & 8 & 9 \\ \hline
\end{tabular}
\endgroup
```

1	2	3
4	5	6
7	8	9

Для наглядности и компактности кода мы ввели и использовали новые команды, задающие цвет линий. Они созданы средствами TEX без проверки возможного наличия команд с такими именами. Но определенные только в пределах группы они не будут конфликтовать с, возможно, уже имеющимися командами.

Отчеркивание средней ячейки сверху и снизу выполнено по-разному. В одном случае линия разбита на отдельные сегменты `\cline`, один из которых сделан красным, а в другом раскрашен один из сегментов линии `\hhline`. Напомним, что последний метод более универсален, так как он будет работать, и если понадобится залить таблицу фоном.

Мы рассмотрели лишь стандартные средства верстки таблиц. Существенно более широкие возможности для этого предоставляют специализированные пакеты tabularray [78] и nicematrix [79], определяющие новые окружения и команды, подробно описанные в их документации.

Глава 11

Указатели

Важной составной частью книг являются указатели, значительно упрощающие поиск нужной информации. Чаще всего составляются предметные или алфавитные указатели, содержащие отсылки к отсортированным по алфавиту понятиям, терминам и т. п., но иногда формируются именные указатели и списки терминов, или *глоссарии*. Приведем пример предметного указателя:

```
\begin{theindex}
.........................
\item лазеры, 12, 39, 115
  \subitem газовые, 87
    \subsubitem CO$_2$, 96
    \subsubitem He-Ne, 89
  \subitem твердотельные, 15
.........................
\end{theindex}
```

Предметный указатель

.....

Л

лазеры, 12, 39, 115
— газовые, 87
— — CO_2, 96
— — He-Ne, 89
— твердотельные, 15

.....

Слева показано окружение theindex, представляющее указатель в рукописи, а справа вид указателя в документе.

Описания элементов указателя помещаются в команды `\index{•}` или `\glossary{•}`. Компилятор LaTeX собирает их при обработке документа. Сортировкой данных и составлением указателя занимается специальные программы, в качестве которых в настоящее время чаще всего выступают программы xindy и xindex. Алгоритмы сортировки

обладают большой гибкостью, позволяющей не только учесть специфику любого алфавита, но наладить сортировку сразу для нескольких языков. Для верстки указателей, имеющих 8-битную кодировку, можно пользоваться программой MakeIndex, подробно описанной в литературе [3,4], но так как она не справляется с сортировкой текста в кодировке unicode, мы ее обсуждать не будем. Мы рассмотрим создание указателя с использованием программы xindex. Она хорошо встраивается в процесс верстки компилятором LaTeX, так как написана на языке Lua. При использовании пакета babel с параметром russian программа xindex автоматически загружает настройки русского языка, и для составления указателя достаточно просто запустить ее, передав ей •.idx файл. Настройки, необходимые для верстки указателей программами xindy и xindex, описаны в прил. А.8 и А.7.

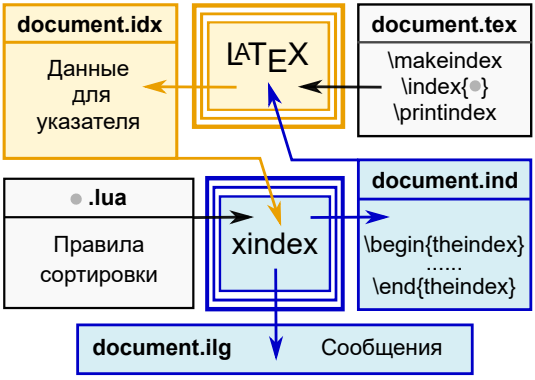


Рис. 11.1. Создание предметного указателя с помощью xindex

Блок-схема процесса генерации предметного указателя показана на рис. 11.1. Стрелками показаны направления потоков информации при взаимодействии LaTeX и xindy. На первом этапе LaTeX собирает записи, находящиеся в рукописи, дополняет их номерами страниц и помещает в файл с расширением •.idx. Затем запускается программа xindex, которая сортирует записи, формирует окружение theindex и записывает его в файл с расширением •.ind. Комментарий о своей работе она направляет на консоль в файл с расширением •.ilg. В ходе последующих компиляций LaTeX вставляет указатель в печатный документ. Рассмотрим, прежде всего, средства LaTeX, формирующие элементы указателя, пригодные для всех программ сортировки.

Для создания указателя нужно загрузить один из пакетов makeidx, [80] imakeidx [81] или xindex [82], налаживающих обработку его элементов. Описывающие их команды \index{*описание элемента*}, LaTeX по умолчанию игнорирует. Сбор элементов необходимо инициировать командой \makeindex, которая должна стоять в преамбуле. Чтобы обеспечить корректность номеров страниц в указателе, команды \index следует помещать непосредственно рядом с терминами. Их можно использовать внутри окружений с рисунками и таблицами. С точки зрения верстки они являются пробелами, поэтому во избежание недоразумений их нужно ставить *после* знаков препинания, скобок и т. д.

При компиляции LaTeX добавляет к описаниям элементов номера страниц, преобразует команды \index в команды

\indexentry{*описание элемента*}{*номер страницы*}

и направляет их в файл с расширением •.idx.

Чтобы указатель появился в печатном документе, рукопись должна содержать команду \printindex, которую LaTeX заменит окружением theindex. При ее отсутствии указатель в документ не вставляется. Окружение theindex представляет собой перечень с вложенностью до трех (xindy) или четырех (xindex) уровней. Записи первого уровня помечаются декларацией \item, а остальных уровней — декларациями \subitem, \subsubitem и \subsubsubitem.[1] В отличие от других списков, в указателе команда \item параметра не имеет. Сортированные записи разбиваются по алфавиту на рубрики.

В описаниях элементов указателя используется синтаксис, разработанный для программы MakeIndex. Описания разбиваются на ряд полей. В простейшем случае они состоят лишь из самого термина, например \index{лазеры}. Поля разных уровней разделяются восклицательным знаком !:

```
... \index{лазеры!газовые} ...
... \index{лазеры!газовые!He-Ne} ...
... \index{лазеры!газовые!CO2@CO$_2$} ...
```

Каждое поле, в свою очередь, может включать в себя дополнительные поля, позволяющие настроить сортировку и выделить среди нескольких ссылок основную. Для сортировки можно определить ключевое слово, которое ставится перед термином и отделяется от него символом @. Например, элемент \index{von Beethoven} попадет в категорию «V», а с ключевым словом (\index{Beethoven@von Beethoven}) он будет

[1] Четвертый уровень указателя и команду \subsubsubitem вводит пакет xindex.

отсортирован по фамилии композитора. Ключевые слова используются в описаниях, содержащих команды LaTeX, например символы, формулы и т. п.: `\index{греческие буквы!alpha@$\alpha$}`. Без ключевого слова формула может вызвать ошибку сортировки и прервать составление указателя. В описаниях можно использовать стандартные команды LaTeX, позволяющие настроить шрифт печати терминов, но они влияют на результат сортировки. Чтобы избежать этого, термины нужно продублировать ключевыми словами. Например, записи второго уровня в представленном выше примере можно набрать курсивом:

```
\index{лазеры!газовые@\textit{газовые}} ...
\index{лазеры!твердотельные@\textit{твердотельные}}.
```

Шрифтовое оформление термина не затрагивает печать номеров страниц. Для ее настройки предназначено еще одно дополнительное поле, следующее за термином и отделяемое от него символом `|`. В простейшем случае оно может содержать имя команды, которая будет использована для вывода номера. Имя приводится без обратного слэша, например номер страницы элемента `\index{лазеры|textbf}` будет жирным. Покажем, как можно изменить шрифты, чтобы сделать более наглядным приведенный выше пример:

```
\index{лазеры|textbf}...
index{лазеры!газовые@\textit{газовые}}...
index{лазеры!газовые!CO2@CO$_2$...
index{лазеры!газовые!He-Ne...
index{лазеры!твердотельные@\textit{...}}...
```

Л
лазеры, **12**, 39, 115
— *газовые*, 87
— — CO_2, 96
— — He-Ne, 89
— *твердотельные*, 15

Круглые скобки, используемые в поле управления выводом ссылки, т. е. `|(` и `|)`, позволяют указать диапазон страниц. Для этого используются две команды. Первая (`\index{`*описание элемента*`|(}`) ставится в начале диапазона, а вторая (`\index{`*описание элемента*`|)}`) — в его конце, при этом *описания* должны быть идентичны. Поставив после `|(` и `|)` имя команды шрифтового выделения, можно настроить шрифт печати страниц диапазона, например `\index{...|(textbf}`.

Вместо номера страницы в ссылку можно внести комментарий:

```
\index{описание элемента|see{текст}}
\index{описание элемента|seealso{текст}}.
```

В первом случае перед *текстом* будет добавлено «*см.*», а во втором — «*см. также*». Комментарий является обычным текстом, чаще всего отсылающим к другому элементу указателя:

```
\index{глоссарий|seealso{указатель}}.
```

Команды `\see{текст}{}` и `\seealso{текст}{}` определены и в обычном тексте, где они используются с двумя аргументами, при этом второй аргумент всегда должен быть пуст. Если в него поместить что-либо, оно пропадет, а если его не указать, пропадет первый символ, следующий за командами. Текст, добавляемый перед комментарием, хранят команды `\seename` и `\seealsoname`. С помощью деклараций `\renewcommand\seename{текст}` и `\renewcommand\seealsoname{...}` его можно изменить.

Заканчивая разбор синтаксиса команды `\index`, добавим важные детали. Если символы `!@|`, служащие для разделения полей, используются как обычные символы, их следует защитить двойной кавычкой `"`, например `\index{"! - факториал}`. То же относится и к самой кавычке: `\index{"" - двойная кавычка}`. Большая проблема возникает при сортировке символов фигурных скобок. Попытки вставить их в указатель командами `\{` и `\}` ведут к ошибкам компиляции. Для их обозначения в командах `\index` пакет xindex вводит команды `\braceLeft` и `\braceRight`, которые не только печатают скобки, но и сортируются как скобки. Еще одна проблема возникает с сортировкой символа `|`, когда в указателе используются гиперссылки. Ее решает замена символа командой `\textbar`.

11.1. Верстка указателей

Рассмотрим теперь верстку указателей с помощью пакета imakeidx [81]. Он позволяет составить сразу несколько указателей, каждому из них присвоить имя и для каждого настроить параметры сортировки и верстки. Для описания элементов и печати указателей он добавляет параметр командам `\index[`*имя указателя*`]{`*описание элемента*`}` и `\printindex[`*имя указателя*`]`. Без параметра они адресуются к указателю, формируемому по умолчанию, не имеющему имени. Параметры относят действие команд к указателям с конкретными *именами*. Данные, используемые для генерации разных указателей, должны быть разделены перед сортировкой. Пакет imakeidx может делать это непосредственно во время компиляции, направляя элементы каждого указателя в индивидуальные файлы. Файлы указателя, формируемого по умолчанию, наследуют имя компилируемого документа и получают расширения •.idx, •.ind и •.ilg. Остальным файлам присваиваются име-

на указателей. Например, если в документе document.tex формируются указатели с именами text, math и без имени, то генерируются файлы document.•, text.• и math.•, где • соответствует всем перечисленным расширениям. Количество файлов, открываемых для записи во время компиляции, ограничено и их может не хватить для создания большого числа указателей. На этот случай предусмотрена еще одна возможность разделения данных, заключающаяся в сохранении одного общего файла и его последующем разделении на компоненты с помощью Perl-скрипта splitindex. В этом случае файл с элементами всех указателей получает имя документа, а имена его компонент составляются из имен документа и указателей, разделенных дефисом. В нашем примере при компиляции будет создан файл document.idx, из которого splitindex и программы сортировки сформируют файлы document-text.•, document-math.• и document-document.•. Использование скрипта splitindex описано в прил. А.9.

Пакет imakeidx способен организовать создание любого числа указателей. Их настройки делятся на индивидуальные и общие. Часть общих настроек устанавливают параметры, перечисляемые через запятую при загрузке пакета.

makeindex | texindy | xindy | truexindy — имя программы, вызываемой для сортировки указателей. По умолчанию считается, что ей является MakeIndex. Синонимы texindy и xindy указывают, что сортировку будет делать программа xindy, адаптированная к синтаксису LaTeX. Параметр truexindy запускает xindy с альтернативными настройками (см. далее и прил. А.8).

splitindex — автоматический запуск скрипта, расщепляющего файл с элементами указателей на компоненты.

noautomatic — отключение автоматического вызова скрипта splitindex и программ сортировки.

quiet — подавление вывода сообщений запускаемых программ.

nonewpage — подавление вывода команды `\newpage`, которая по умолчанию ставится перед каждым указателем в классе article.

original — использование окружения theindex, определенного классом компилируемого документа, без дополнений пакета imakeidx.

В списке программ, поддерживаемых пакетом imakeidx, пока отсутствует xindex, но это упущение устраняет пакет xindex. Он автоматически загружает imakeidx и назначает xindex сортирующей программой.

Команда `\indexsetup{`*список настроек*`}` формирует второй набор общих настроек. Они перечисляются через запятую в ее аргументе.

headers = {*левый колонтитул*}{*правый колонтитул*} — установка колонтитулов страниц с указателями.

firstpagestyle = *стиль страницы* — установка стиля первой страницы указателей. По умолчанию используется plain.

noclearpage — подавление команды `\clearpage`, по умолчанию выполняемой между указателями.

toclevel = *имя команды разделов* — установка уровня указателей в оглавлении, например `toclevel = section`. По умолчанию указатели являются главами.

level = *команда разделов* — стиль вывода заголовков указателей. По умолчанию используется `\chapter*`. Команда с именем без звездочки автоматически вставляет указатели в оглавление.

othercode = *набор команд* — дополнительные команды, выполняемые перед выводом элементов указателей. Обычно они настраивают параметры верстки, например `othercode=\small` уменьшит шрифт всех указателей.

Одним из важных достоинств пакета imakeidx является возможность индивидуальной настройки указателей, обеспечиваемая командами `\makeindex[`*список настроек*`]` . Каждому указателю можно присвоить имя и составить собственный список настроек.

name = *имя* — присваивает указателю имя, используемое для генерации файлов и адресации других команд. Если данный параметр не задан, настройки применяются к указателю, формируемому по умолчанию.

title = {*заголовок*} — устанавливает текст заголовка указателя. Если параметр отсутствует, используется заголовок, хранящийся в команде `\indexname`.

program = **makeindex** | texindy | xindy | truexindy — устанавливает программу сортировки, если она не указана при загрузке пакета, или отличается от указанной.

options = *список ключей* — устанавливает список ключей, передаваемый программе сортировки (см. прил. А.7 и А.8).

noautomatic = true | **false** — подавляет используемый по умолчанию автоматический вызов программы сортировки.

intoc = true | **false** — вносит указатель в оглавление, если параметр intoc не задан при загрузке пакета, в противном случае исключает его из оглавления.

columns = *число* — задает количество колонок в указателе, по умолчанию равное двум.

columnsep = *длина* — задает размер интервала между колонками, по умолчанию равный 35pt.

columnseprule — разделяет колонки вертикальной линией.

Приведем пример команды, формирующей англо-русский указатель с именем «pkg» и заголовком «Пакеты», сортируемый программой xindex без учета регистра букв (см. прил. А.7):

```
\makeindex[name= pkg, title= Пакеты, options= -a -c RU -l RU].
```

Каждый указатель можно снабдить введением, которое выводится после заголовка, предваряя список элементов. Для этого определена команда `\indexprologue[`*вертикальный пробел*`]{`*текст*`}`. Ее аргументом является один или несколько абзацев текста, а в параметре можно задать вертикальный пробел, отделяющий введение от элементов указателя. По умолчанию используется пробел `\bigskip`. Собственное введение есть в каждом указателе данной книги. Например, в представленном на с. 484 указателе «Пакеты», его формируют команды

```
\indexprologue{ \label{p:Packages}
   Символы \texttt{[•]} обозначают наличие настроек,...}
\printindex[pkg]
```

Метка в команде `\indexprologue` позволяет сослаться на страницу с указателем.

Команды `\indexprologue` и `\printindex` работают в паре. Вставив введение, команда `\printindex` аннулирует его, чтобы оно не появилось в следующем указателе, поэтому перед его выводом, если требуется, введение нужно сформировать заново командой `\indexprologue`.

Суммируем кратко информацию о настройке указателей пакетом imakeidx. Команда `\indexsetup` формирует колонтитулы (headers), устанавливает стиль оформления элементов (overcode) и заголовков (level), а также задает уровень указателей в оглавлении (toclevel). Команды `\makeindex` вносят указатели в оглавление (intoc), формируют текст их заголовков (title), устанавливают число колонок (columns) и расстояние между ними (columnsep). Эти команды должны находиться в преамбуле документа. Команды `\indexprologue`, формирующие введение указате-

лей, лучше располагать непосредственно перед их печатью командами `\printindex`.

Пакет imakeidx обеспечивает автоматический вызов программ, необходимых для составления указателей, непосредственно во время компиляции. Это не требуется, если формируется один указатель, так как оболочка TeXstudio способна сама настроить и запустить программу сортировки по окончании компиляции. Проблема возникает при генерации нескольких указателей, в которой может участвовать несколько программ с разными настройками. В этом случает помощь пакета imakeidx очень существенна, но она реализуется, только если компилятор LaTeX предоставит ему возможность запускать внешние программы, обычно отключаемую в целях безопасности. Для ее активации в строке запуска компиляторов следует добавить ключ `-shell-escape` в ОС Linux, или `--enable-write18` в ОС Windows. В программе TeXstudio список ключей pdflatex редактируется в меню «Options ⇒ Configure TeXstudio... ⇒ Commands ⇒ PdfLaTeX».

11.2. Сортирующие программы

Сортируют и формируют указатели программы MakeIndex, xindex и xindy. С помощью специфического набора команд, описанных в литературе [3, 4], для программы MakeIndex можно составить стилевые файлы, гибко настраивающие все аспекты сортировки и верстки. Хотя, как упоминалось ранее, программа MakeIndex не работает с файлами кодировки unicode, в принципе, данные, собранные при компиляции, можно перекодировать в 8-битную кодировку, составить указатели с помощью MakeIndex, а затем перекодировать их обратно в unicode. Этот путь неудобен и чреват появлением ошибок при перекодировке файлов, поэтому от услуг программы MakeIndex лучше отказаться в пользу xindex или xindy.

Долгое время основным инструментом формирования unicode-кодированных указателей была программа xindy. Разработчики хорошо адаптировали ее не только к специфическому синтаксису команд `\index`, но и к синтаксису LaTeX. Она успешно справляется с сортировкой списков, однако верстка окружения theindex хорошо отлажена только для отдельных языков. Составление двуязычных указателей, например англо-русских, требует особых настроек, не предусмотренных ее стандартными средствами. К сожалению xindy не работает в ОС

Windows, а в ОС Linux она запускается только в дистрибутиве TeXLive. Работа с программой xindy обсуждается в прил. А.8.

В последние годы активно развивается программа xindex. Она работает во всех операционных системах и прекрасно справляется с составлением англо-русских указателей, используя лишь собственные средства. Для этого ее нужно запустить с ключами `-c` и `-l`:

```
xindex -c RU -l RU •.idx
```

Первый загружает конфигурационный файл, второй устанавливает русские заголовки рубрик «Символы» и «Числа». Конфигурационный файл настраивает сортировку, а также задает отступы и символы, выводимые перед элементами различных уровней, шрифт, которым печатаются заголовки рубрик, и некоторые другие параметры верстки. Полный список настроек и описание конфигурационного файла приведены в прил. А.7. Номера страниц в указателях можно оформить в виде гиперссылок. Для этого достаточно загрузить пакет hvindex [83].

Программа xindex сортирует элементы указателя посимвольно. Например элементы `\index{\texttt{\textbackslash command}}` ⇝ `\command` и `\index{\textit{example}}` ⇝ *example*, первым символом которых является «\», она поместит в рубрику символов, поставив команду `\command` после *example*, так как в `\texttt` за `\text` следует буква «t», тогда как в `\textit` — «i». Если же добавить в записи ключевые слова, элементы попадут в разные рубрики:

```
\index{example@\textit{example}},
\index{command@\texttt{\textbackslash command}},
```

C
`\command`, …
E
example, …

Мы рекомендуем всегда придерживаться строгих правил и предварять поля, содержащие команды LaTeX. ключевыми словами,

11.3. Списки сокращений и терминов

Список терминов, т. е. *глоссарий*, и список сокращений — *аббревиатур*, удобней всего верстать в окружении description, воспользовавшись механизмом формирования указателей.

Одну из возможностей формирования глоссариев, используем это общее название и для списков аббревиатур, предоставляют пакет makeidx и программа MakeIndex. В ядре LaTeX определена команда `\glossary{`*описание элемента*`}`, идентичная команде `\index{•}`.

Пакет makeidx обеспечивает обработку этих команд, если она инициирована в преамбуле рукописи декларацией `\makeglossary`. Тогда компилятор преобразует их в команды

`\glossaryentry{`*описание элемента*`}{`*номер страницы*`}`

и направляет в файл с расширением •.glo. Для сортировки и верстки глоссариев можно использовать специальные стилевые файлы программы MakeIndex, примеры которых приведены в книгах [3,4]. Сверстанные таким образом глоссарии обычно представляют собой окружения theindex, помещенные в файлы с расширением •.gls. Команда, автоматически вставляющая их в документ, не предусмотрена, поэтому •.gls-файлы придется загружать «вручную» с помощью команды `\input`. Как и в случае указателей, составляя глоссарии, от помощи программы MakeIndex лучше отказаться.

Богатые ресурсы верстки глоссариев предлагает пакет glossaries. Чтобы обеспечить их, он вводит более сотни новых команд, подробно описанных на семистах девяти страницах его документации [84]. Заинтересованный читатель может ознакомиться с данными ресурсами из краткого введения, содержащего «всего» пятьдесят семь страниц [85], мы же опишем более простой путь, в большинстве случаев обеспечивающий приемлемый результат.

Верстка глоссариев отличается от верстки указателей. Указатели обычно разбиваются на рубрики, и в начале каждой их них выводится большая буква-заголовок, что вряд ли требуется в глоссариях. Описания терминов или аббревиатур не должны отсылать читателя к страницам, на которых они встречаются, как это делают элементы указателей. Объемный текст расшифровки термина плохо смотрится при верстке в несколько колонок, применяемой для вывода указателей. Размер шрифта в глоссариях обычно такой же, как и у основного текста, тогда как в указателях шрифт уменьшается. Указатели верстает окружение theindex, элементы которого печатаются одинаковым шрифтом, но разбиваются на уровни, зрительно разделяемые между собой. В глоссариях это не представляется столь необходимым. Для их верстки более подходит окружение description, все записи которого печатаются одинаково, а формирующие их команды `\item` имеют параметр, позволяющий выделить термин из его описания.

Оптимальный набор средств формирования глоссариев предоставляют пакет imakeidx и программа xindex. Рассмотрим для примера верстку списка сокращений и списка терминов, начав с наиболее простого.

Список сокращений

ARPES (Angle resolved photo electron spectroscopy) — фотоэлектронная спектроскопия с угловым разрешением.
EXAFS (Extended x-ray absorption fine structure) — протяженная тонкая структура рентгеновского спектра поглощения.
HTSC (High temperature superconductivity) — высокотемпературная сверхпроводимость.
MOCVD (Metalorganic chemical vapour deposition) — осаждение металлоорганических соединений из газовой фазы.
PES (Photo emission spectroscopy) — фотоэлектронная спектроскопия.
PLD (Pulsed laser deposition) — импульсное лазерное напыление.
SCES (Strongly correlated tlectron systems) — системы с сильными электронными корреляциями.
XAFS (Extended x-ray absorption fine structure) — тонкая структура рентгеновского спектра поглощения.
XANES (X-ray absorption near edge structure) — околопороговая тонкая структура рентгеновского спектра поглощения.
XAS (X-ray absorption spectroscopy) — спектроскопия рентгеновского поглощения.

```
\makeindex[name=abbrv, options= -c abbreviations -l RU -g -a -n]

\begin{document}

\section*{Список сокращений}
\input{abbrv.ind}

... в системах с сильными электронными корреляциями (SCES)
\index[abbrv]{SCES (Strongly correlated tlectron systems) ---
системы с сильными электронными корреляциями.} ...

%\printindex[abbrv]
```

Рис. 11.2. Список аббревиатур и формирующий его набор команд

Пример списка аббревиатур показан на рис. 11.2. Чтобы сверстать его, в преамбуле рукописи нужно инициировать сбор и сортировку аббревиатур командой `\makeindex`. В нашем случае она формирует список с именем abbrv. Расшифровки аббревиатур можно разместить в тех местах текста, где они вводятся. Команда `\printindex[abbrv]` должна стоять в конце рукописи, так как после нее обработка команд `\index[abbrv]{...}` прекратится, и вводимые за ней аббревиатуры в список не попадут. Она автоматически напечатает список в конце

документа, но в некоторых случаях, например в отчетах о НИР, оформляемых по ГОСТ 7.32-2017, он должен находиться в начале. Чтобы вывести его там, используется команда `\input`.

Два одинаковых списка в документе не нужны, поэтому команду `\printindex[abbrv]` после формирования списка следует закомментировать, а если понадобится добавить новые аббревиатуры, ее можно раскомментировать вновь. Перед *начальным* созданием списка нужно закомментировать команду `\input`, иначе загрузка несуществующего файла прервет компиляцию документа.

Программа сортировки запускается с настройками

```
xindex -n -g -a -c abbreviations -l RU abbr.idx.
```

Ключ `-c` загружает описанный в прил. А.7 конфигурационный файл xindex-abbreviations.lua, предназначенный для верстки аббревиатур. Ключи `-n` и `-g` предотвращают вывод заголовков рубрик и номеров страниц. Ключ `-a` убирает различие заглавных и строчных букв при сортировке. Как и в случае указателей, сортируемые записи можно разделить на четыре уровня, однако после сортировки xindex заключит их в окружение description, где все они окажутся на одном уровне.

Команда `\section*` оформляет заголовок списка аббревиатур. В нашем примере «раздел» со списком не нумеруется и в оглавление не вставляется. Если это потребуется, воспользуйтесь командой `\addcontentsline` (см. разд. 8.5).

Декларации `\item[•]`, формирующие записи окружения description, имеют параметр, не использованный для верстки аббревиатур. Мы применим его в верстке глоссария, представленного на рис. 11.3. Рубрики глоссария формируют команды `\index[glos]{Ф@[Фермионы]}` и `\index[glos]{Б@[Бозоны]}`. При сортировке ключи `Б` и `Ф` разводят термины `[Фермионы]` и `[Бозоны]` по разным рубрикам. Текст, заключенный в квадратные скобки, после обработки программой xindex становится параметром команд `\item`. По умолчанию он выделяется жирным шрифтом.

Списки частиц в рубриках составляются из команд типа

```
\index[glos]{Ф@[Фермионы]!Фермион@[\sf Фермион]...},
\index[glos]{Ф@[Бозоны]!Бозон@[\sf Бозон]...}.
```

Названия частиц, находящиеся в квадратных скобках, тоже становятся параметрами команд `\item`. Декларации `\sf` печатают их рубленым шрифтом. Названия, находящиеся в поле ключей, гарантируют правильную сортировку частиц.

Список терминов

Элементарные частицы — первичные, далее неразложимые частицы, из которых, предположительно, состоит вся материя. Они подчиняются квантовой статистике и с этих позиций делятся на *бозоны* и *фермионы*.

Бозоны

Бозон — частица с нулевым или целым спином S.

Мезон — нестабильная частица класса адронов: $S = 0$.

Пи-мезоны — группа нестабильных сильно взаимодействующих нейтральных (π^0), положительно (π^+) или отрицательно (π^+) заряженных частиц, обладающих массой, промежуточной между массами *протона* и *электрона*: $S = 0$.

Фотон — квант электромагнитного излучения: $S = 1$.

Фермионы

Нейтрино — стабильная легкая (возможно безмассовая) незаряженная частица: $S = 1/2$.

Нейтрон — незаряженная частица с большим временем жизни: $S = 1/2$.

Позитрон — стабильная положительно заряженная частица, античастица *электрона*: $S = 1/2$.

Протон — стабильная положительно заряженная частица: $S = 1/2$.

Фермион — частица с полуцелым спином S.

Электрон — стабильная отрицательно заряженная частица: $S = 1/2$.

```
\makeindex[name=glos, options= -c description -l RU -a -n -g]

\begin{document}

\section*{Список терминов}
\textit{Элементарные частицы} --- первичные,...

\medskip
\input{glos.ind}

... \index[gls]{Ф@[Фермионы]}
\index[glos]{Ф@[Фермионы]![\sf Фермион] --- частица с полуцелым...}

... \index[gls]{Б@[Бозоны]}
\index[glos]{Б@[Бозоны]![\sf Бозон] --- частица с нулевым или...}

%\printindex[glos]
```

Рис. 11.3. Глоссарий и формирующий его набор команд

Программа сортировки запускается с настройками

```
xindex -n -g -a -c description -l RU glos.idx.
```

Используемая конфигурация сортировки xindex-description.lua, обсуждаемая в прил. А.7, отличается от конфигурации xindex-abbreviations.lua тем, что в списке терминов записи разделяет небольшая отбивка, ненужная в списке аббревиатур.

В нашем примере перед глоссарием выводится заголовок и преамбула, которая является обычным текстом. Пробел `\medskip` отделяет ее от глоссария. Печатью глоссария управляют настройки окружения description. Измените их, если потребуется какой-то специальный вид списка терминов.

11.4. Резюме

Подведем итог краткого обзора формирования указателей, глоссариев и списков аббревиатур. Тандем пакета imakeidx и программы xindex дает компактный и удобный набор средств, гибко настраивающих сортировку элементов и практически все аспекты верстки этих списков. Необходимые настройки программы xindex подробно описаны в прил. А.7.

Глава 12

Замена и настройка шрифтов

В разд. 2.6 мы обсудили управление шрифтами на уровне пользователя. Рассмотренные команды, настраивающие атрибуты шрифтов, от самих шрифтов не зависят. Гарнитуры Computer Modern, по умолчанию используемые LaTeX, обычно отличаются от шрифтов, которыми верстается манускрипт в издательстве. Чтобы точнее оценить объем статьи и соблюсти все требования к ее оформлению, некоторые издательства рекомендуют заменять шрифты на те, что будут использованы в издательском процессе. Для этого предназначены специальные шрифтовые пакеты. При правильном подходе такая замена не затрагивает другие аспекты верстки.

По умолчанию LaTeX настроен на работу с растровыми шрифтами, глифы которых состоят из набора точек. По мере необходимости программа METAFONT генерирует такие шрифты «на лету». Для верстки pdf-документов растровые шрифты лучше заменить векторными, глифы которых отрисованы кривыми. Компилятор pdfTeX работает с векторными шрифтами форматов PostScript и TrueType. На их основе был создан формат OpenType, обладающий более широкими возможностями и высокой компактностью.

Шрифтовые технологии, используемые компилятором pdfTeX, стремительно уходят в прошлое, поэтому не останавливаясь на PostScript гарнитурах, с которыми он работает, в данной главе мы рассмотрим шрифты OpenType, которыми оперируют компиляторы XeTeX и LuaTeX.

Мы начнем с обсуждения их подключения и сделаем обзор ряда гарнитур, поддерживающих верстку на основных европейских языках. Затем мы разберем механизм оперирования шрифтами на уровне ядра LaTeX, так называемую схему NFSS, и опираясь на нее, опишем настройку OpenType гарнитур.

12.1. Замена шрифтов

Одним из главных преимуществ компиляторов нового поколения является *непосредственное* использование кодировки unicode и OpenType шрифтов, установленных в операционной системе или в дистрибутиве TeX. Они пока еще не рекомендованы для подготовки публикаций научной периодики, но могут с успехом применяться в верстке документов «внутреннего пользования». Мы применим общее наименование OpenType и к формату TrueType, послужившему основой OpenType технологий.

В используемой сейчас 16-битной адресации шрифты OpenType вмещают до 65536 символов, т. е. адресных кодов, при этом количество глифов, т. е. графических представлений символов, может быть еще больше, так как, в отличие от обсуждавшихся ранее 8-битных шрифтов, символы OpenType шрифтов могут иметь несколько вариантов начертаний, например обычные и вычурные буквы вместе с капителью, несколько стилей для каждого варианта и даже цветные версии. Помимо глифов шрифты содержат в себе всю информацию, необходимую для работы с ними, типа метрик, лигатур и т. п. Как правило, OpenType шрифты вмещают несколько скриптов, т. е. наборов глифов, печатающих определенный набор символов (кодов).

Хорошо проработанные шрифты насчитывают несколько оптических размеров (кеглей). Обычно их выделяют в отдельные файлы, но технология OpenType позволяет работать с ними как с единым шрифтом. В то же время большинство шрифтов разрабатывается только в одном размере, из которого масштабированием получают все требуемые. Такие шрифты обеспечивают хорошую компактность документов, но набранный ими текст комфортен для глаз лишь в ограниченном диапазоне размеров: очень мелкие читаются с трудом, а очень крупные выглядят чересчур контрастно. Тем не менее при правильном подборе основного кегля масштабируемые шрифты обеспечивают вполне приемлемое качество оформления основных элементов документа от заголовков до сносок.

Шрифты OpenType используют кодировку unicode, но чтобы обрабатывать команды, заменяющие символы в рукописи и файлах, генерируемых в ходе компиляции, перед версткой загружается кодировка TU, вмещающая кодировки TS1 и T1. В ней отсутствуют команды, печатающие буквы кириллицы, и набранный ими документ будет компилироваться с ошибками. Поэтому при переходе от компилятора PDFLaTeX к LuaLaTeX или XeLaTeX следует удалить файлы, автоматически сгенерированные ранее, и заменить буквы-команды символами. Кроме этого в преамбуле документа нужно заменить пакеты inputenc, fontenc и cmap пакетом fontspec или unicode-math (см. далее).

Чтобы компилировать документ с OpenType шрифтами в программе TeXstudio, откройте меню «Tools ⇒ Commands» и выберите «LuaLaTeX» или «XeLaTeX», а чтобы работать с ними постоянно, замените компилятор, используемый для сборки документа. Для этого, зайдя в меню «Options ⇒ Configure TeXstudio...», откройте меню «Build» и, нажав на стрелочку выпадающего списка «Default Compiler», выберите компилятор `xts:///lualatex` или `xts:///xelatex`.

12.1.1. Установка OpenType шрифтов

Шрифты лучше устанавливать в ресурсы операционной системы, тогда они будут доступны всем приложениям, включая компиляторы XeTeX и LuaTeX. Если установить шрифт в ОС не удается, добавьте его в дистрибутив TeX.

MiKTeX не устанавливает шрифты автоматически при первом обращении к ним. Чтобы установить шрифты вручную, откройте MiKTeX Console, выберите меню Packages, введите название содержащего их пакета в строку поиска и найдите его. Нажав на строку с найденным пакетом правой клавишей мыши, установите шрифты, выбрав «Install package» во всплывающем меню. Чтобы убрать шрифты из дистрибутива, проделайте те же действия, выбрав в конце «Remove package».

Некоторые шрифты автоматически устанавливаются пакетами, загружаемыми командой `\usepackage`. Они конфигурируют шрифты и, чтобы задействовать все их ресурсы, зачастую определяют ряд *нестандартных* команд. Далее мы будем лишь упоминать об этом, отсылая заинтересованного читателя к документации пакетов, в которой они описаны.

12.1.2. Замена текстовых шрифтов

Для подключения OpenType шрифтов необходимо загрузить пакет fontspec [86] и задать гарнитуры декларациями

```
\setmainfont[настройки]{гарнитура}[настройки],
\setsansfont[настройки]{гарнитура}[настройки],
\setmonofont[настройки]{гарнитура}[настройки].
```

Первая устанавливает основной шрифт, обычно соответствующий гарнитуре Serif, вторая и третья — шрифты гарнитур SansSerif и Mono. Хотя оба параметра деклараций равноценны, разработчик пакета рекомендует помещать *настройки* в последний.

Названия *гарнитур* могут отличаться от названий файлов со шрифтами. Например, в ОС Windows название файлов, содержащих гарнитуру Times New Roman, и расположение папки, в которой они хранятся, зависит от версии ОС. Чтобы узнать название незнакомой гарнитуры, откройте консоль в папке со шрифтом и запустите программу

otfinfo -i *имя файла*,

устанавливаемую дистрибутивами TEX. Она выведет информацию о шрифте, в которой название гарнитуры содержит поле (строка) Preferred family, а если его нет, то поле Family.

Размер основного шрифта устанавливается в соответствии с кеглем, заданным классом документа, а потом умножается на масштабирующий коэффициент, определяемый в *настройках* с помощью параметра

Scale = *значение*.

Значением может быть десятичное число или специальные параметры MatchUppercase и MatchLowercase, автоматически выравнивающие высоту заглавных или строчных букв с основным шрифтом, а также параметр MatchAveragecase, устанавливающий среднее значение двух предыдущих параметров. Размеры гарнитур SansSerif и Mono привязывают к основному шрифту, а сам основной шрифт выравнивают с используемой по умолчанию гарнитурой Latin Modern Roman — OpenType, аналогом шрифтов Computer Modern. Приведем пример, в котором декларации активируют гарнитуры Times New Roman, Arial и Courier New, выравнивая строчные буквы первой и второй:

```
\usepackage{fontspec}
\setmainfont{Times New Roman}
\setsansfont{Arial}[Scale=MatchLowercase]
\setmonofont{Courier New}
```

При подключении гарнитуры, как правило, автоматически доступны прямые, курсивные или наклонные шрифты и капитель обычной и жирной насыщенности. Если шрифты содержат цифры старого стиля, команда `\oldstylenums{•}` печатает ими числа.

12.1.3. Замена математических шрифтов

Чтобы документ выглядел гармонично, стиль и контрастность текста и формул должны быть согласованы между собой, поэтому качественные гарнитуры содержат текстовые и математические шрифты, разработанные в едином стиле.

Математические шрифты в формате OpenType подключает пакет unicode-math. Хотя в настоящее время ему присвоен экспериментальный статус [87], он полностью функционален и поддерживает более десятка математических гарнитур. Пакет открывает доступ к тысячам математических символов, которые можно вводить в формулы как с помощью команд, так и непосредственно в виде символов [88]. Предваряя более подробное обсуждение верстки формул в разд. 12.3.6, отметим, что пакет unicode-math автоматически загружает пакет amsmath.

Пакеты fontspec и unicode-math тесно связаны между собой. Последний нуждается в первом и сам загружает его, если это не было сделано ранее.

Пакет unicode-math очень прост в использовании. Его настройки, применяемые по умолчанию, соответствуют стилю верстки формул стандартного LaTeX, поэтому достаточно загрузить его без параметров и задать математическую гарнитуру декларацией

`\setmathfont{`*гарнитура*`}[`*настройки*`]`,

имеющей такой же синтаксис, что и команды, устанавливающие гарнитуры текстовых шрифтов. В частности, параметром Scale можно выровнять высоту математических шрифтов с текстовыми. Остальные параметры обсуждаются в разд. 12.3.6. Если опустить декларацию `\setmathfont`, формулы будут верстаться OpenType шрифтом Latin Modern Math, аналогом шрифтов Computer Modern. Согласно ГОСТу и российским полиграфическим традициям греческие буквы в формулах должны набираться прямым шрифтом. Если гарнитура содержит такие шрифты, их можно активировать с помощью двух деклараций:

```
\setmathfont{гарнитура}
\setmathfont{гарнитура}[range=it/greek->up,bfit/greek->bfup]
```

Первая задаст гарнитуру, а вторая заменит курсивные греческие буквы прямыми (см. разд. 12.3.6). Обе команды обязательны, даже если гарнитура одна и та же.

12.1.4. Примеры шрифтов

В настоящее время имеется большое количество свободно распространяемых OpenType шрифтов, доступных для верстки в LaTeX: `https://tug.org/FontCatalogue/allfonts.html`. Их полиграфическое качество сильно различается и большинство из них содержит лишь алфавиты на основе латиницы. Для примера в табл. 12.1 показаны профессионально изготовленные гарнитуры, в равной мере поддерживающие латиницу и кириллицу.

Обозначения Капитель и SmallCapitals означают, что в первом случае шрифты поддерживают капитель латиницы и кириллицы, а во втором только латиницы, кириллица же печатается строчными буквами. *Наклоненные* шрифты гарнитур New Computer Modern генерируются посредством трансформации матрицы прямого шрифта, см. разд. 12.3.4. Шрифты, выделенные фоном, доступны в свободно распространяемой версии v.2.82 гарнитуры Times New Roman. Ее более поздние версии содержат также капитель и вычурные шрифты. Последние не активируются автоматически и потому не внесены в таблицу. Их подключение описано в разд. 12.3.5.

Гарнитуры Computer Modern

Выбор гарнитур Computer Modern в формате OpenType довольно широк (табл. 12.1). Дистрибутивы TeX устанавливают гарнитуры Latin Modern, содержащие шрифты разных оптических размеров (Roman от 5pt до 17pt, SansSerif от 8pt до 17pt, Typewriter от 8pt до 12pt), наиболее полно воспроизводящие гарнитуры Computer Modern. Для работы с ними пакет fontspec не требуется, компиляторы LuaLaTeX и XeLaTeX используют их по умолчанию. При этом формулы верстаются математическими шрифтами Computer Modern, а если загрузить пакет lmodern, для печати букв будут использованы шрифты Latin Modern. Если дополнительно загрузить пакет unicode-math, не указав явно математический шрифт, формулы будут верстаться гарнитурой Latin Modern Math, которая будет установлена автоматически. Данная гарнитура сделана в единственном размере, поэтому мелкие детали формул с ней смотрятся хуже, чем в формулах, сверстанных шрифтами Computer Modern.

Таблица 12.1

Шрифты TrueType и OpenType гарнитур

Гарнитуры	Шрифты
NewCM10-Regular	Прямой, *наклоненный*, *курсив*, **прямой, *наклоненный*, *курсив*,** Капитель, **Капитель.**
Times New Roman	Прямой , Капитель, *курсив* , *Капитель*, **прямой , Капитель, *курсив* , *Капитель*.**
XITS	Прямой, SmallCapitals, *курсив*, **прямой, *курсив*.**
STIX Two Text	Прямой, Капитель, *курсив, Капитель*, **прямой, Капитель, *курсив, Капитель*.**
EBGaramond	Прямой, SmallCapitals, *курсив, SmallCapitals*, *Swash, SmallCapitals*, **прямой, SmallCapitals, *курсив, SmallCapitals*, *Swash, SmallCapitals*.**
NewCMSans10-Regular	Прямой, *наклонный*, SmallCapitals, **прямой, *наклонный*.**
Calibri	Прямой, Капитель, *курсив, Капитель*, **прямой, Капитель, *курсив, Капитель*.**
Lato	Прямой, *курсив*, **прямой, *курсив*.**
NewCMMono10-Regular	Прямой, *наклоненный*, *курсив*, SmallCapitals, **прямой, *наклоненный*.**
Courier New	Прямой, *наклонный*, **прямой, *наклонный*.**
Anonymous Pro	Прямой, *наклонный*, **прямой, *наклонный*.**

К сожалению, гарнитуры Latin Modern не поддерживают кириллицу. При компиляции русские буквы бесследно исчезают, поэтому в документах, написанных по-русски, нужно использовать другие шрифты. Пакет cm-unicode предоставляет коллекцию шрифтов CM Unicode, поддерживающую латиницу и кириллицу. Ее гарнитуры содержат полный набор начертаний и контрастностей, но шрифты разработаны лишь в одном размере, что снижает их полиграфическое качество. Гарнитуры CM Unicode нужно установить вручную, а затем подключить с помощью пакета fontspec:

```
\usepackage{fontspec}
\setmainfont{CMU Serif}
\setsansfont{CMU Sans Serif}
\setmonofont{CMU Typewriter Text}.
```

Еще одни шрифты — New Computer Modern, поддерживающие латиницу, кириллицу и греческий алфавит, автоматически устанавливает пакет fontsetup. Их гарнитуры Serif и SansSerif разработаны в оптических размерах 8pt и 10pt. Первый используется для мелких кеглей, второй — для шрифтов размером от 9pt. Гарнитура Mono сделана только в размере 10pt. В коллекции имеются и математические гарнитуры New Computer Modern Math, разработанные в размере 10pt, содержащие две тысячи триста сорок символов.

Гарнитуры New Computer Modern имеют три варианта контрастности. Шрифты Regular, обладающие контрастностью шрифтов Computer Modern, хорошо смотрятся на мониторе. Чуть более контрастные шрифты Book предназначены для печати. С гарнитурами Regular и Book комбинируются общие жирные шрифты Bold.

Гарнитуры New Computer Modern [89] автоматически подключает пакет newcomputermodern, входящий в коллекцию пакетов fontsetup:

```
\usepackage[regular]{newcomputermodern}
\setmathfont{NewComputerModernMath}[range=it/greek->up,bfit/greek->bfup].
```

При загрузке без параметров используются шрифты Book, а с параметром regular более светлые шрифты Regular. Шрифты можно подключить и, воспользовавшись пакетом fontsetup:

```
\usepackage[olddefault]{fonsetup}
```

Его загрузка без параметров или с параметром default активирует шрифты контрастности Book, а с параметром olddefault — контрастности Regular. Другие параметры загрузки, описанные в документации fonsetup [90], настраивают вывод некоторых символов и ак-

Шрифты New Computer Modern

Рубленые и моноширинные шрифты должны хорошо выделяться из окружающего текста и при этом гармонировать с ним. Они могут быть чуть крупнее или мельче основного шрифта и отличаться контрастностью.

Стоит помнить, что текст, выделенный изменением *начертания* или **контрастности** самого́ основного шрифта, обычно смотрится лучше. Его составляющие сбалансированы оптимально, и чтение минимально утомляет глаза.

`\oldstylenums` ⇝ 1234567890, 1234567890, 1234567890.

{\tiny Пример текста, сверстанного крошечным шрифтом 5pt. Гарнитура Roman. Гарнитура SansSerif. Гарнитура Typewriter.}

Документ выглядит гармонично, если шрифты текста и формул разработаны в едином стиле, как в данном примере. В «строчных» формулах индексы располагаются сбоку от знака суммирования, а в «вынесенных» — над и под ним. Обратите внимание на отличие выражения $\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!}$ от его аналога:

$$\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = \lim_{x\to 0} (1+x)^{1/x} = 2.718281828\ldots$$

В некоторых формулах, например $\lim\limits_{x\to 0} (\mathrm{e}^x - 1)/x = 1$, индексы лучше опустить вниз, а для разделения числителей и знаменателей дробей использовать обычный слэш.

В качестве еще одного примера проинтегрируем вектор Пойнтинга по малому интервалу времени δt и объему магнетика:

$$-\int \mathbf{jE}\,\delta t\,dV = \frac{c}{4\pi}\int \nabla(\mathbf{E}\times\mathbf{H}_m)dV\delta t + \frac{1}{4\pi}\int\left(\mathbf{E}\frac{\partial\mathbf{D}}{\partial t} + \mathbf{H}_m\frac{\partial\mathbf{B}}{\partial t}\right)\delta t\,dV = \\ = \frac{c}{4\pi}\oint(\mathbf{E}\times\mathbf{H}_m)d\mathbf{S}\,\delta t + \frac{1}{4\pi}\int(\mathbf{E}\delta\mathbf{D} + \mathbf{H}_m\delta\mathbf{B})dV.$$

Для обозначения переменных используются математические алфавиты и греческие буквы:

$ABCDEFGHIJKLMNOPQRTSUVWXYZ$

$abcdefghijklmnopqrtsuvwxyz$

$\Gamma\Delta\Theta\Lambda\Xi\Pi\Sigma\Phi\Psi\Omega$ $\alpha\beta\gamma\delta\varepsilon\zeta\eta\vartheta\iota\kappa\lambda\mu\nu\xi\pi\rho\varrho\sigma\tau\upsilon\phi\varphi\chi\psi\omega$

$\mathfrak{ABCDEFGHIJKLMNOPQRTSUVWXYZ}$ $\mathfrak{abcdefghijklmnopqrtsuvwxyz}$

$\mathbb{ABCDEFGHIJKLMNOPQRTSUVWXYZ}$ $\mathbb{abcdefghijklmnopqrtsuvwxyz}$

$\mathcal{ABCDEFGHIJKLMNOPQRTSUVWXYZ}$

Рис. 12.1. Образец шрифтов New Computer Modern

тивируют ряд других шрифтов. В частности, дополненный пакетом fontsetup-nonfree,[1] пакет fontsetup позволяет работать с коммерческими шрифтами Cambria, MS-Garamond, Lucida, MinionPro и Linotype Palatino.

Настраивая шрифты, пакеты fontsetup и newcomputermodern автоматически загружают пакеты fontspec и unicode-math и подключают для верстки формул гарнитуру New Computer Modern Math нужной контрастности, поэтому в ее явном объявлении нужды нет. В представленной ранее загрузке декларация `\setmathfont` заменяет курсивные греческие буквы прямыми, если ее опустить, формулы будут верстаться в стандартном стиле TEX, в котором строчные греческие буквы набираются курсивом, а прописные прямым шрифтом.

Гарнитуры Serif и Mono шрифтов New Computer Modern содержат *наклоненные* шрифты, генерируемые из прямого шрифта с помощью технологий OpenType. Качество таких шрифтов хуже, чем специально разработанных *наклонных* шрифтов, имеющихся в гарнитурах CM Unicode. Из взаимодополняющих шрифтов можно составить новую гарнитуру, собрав в ней наиболее качественные шрифты. В разд. 12.3.5 описано, как это сделать.

Пример гарнитур New Computer Modern и Latin Modern Math показан на рис. 12.1. Один из фрагментов его текста набран мелким шрифтом. В гарнитурах, разработанных в одном оптическом размере, такой текст читается с трудом. Сравните текст, сверстанный шрифтами `\tiny` гарнитур New Computer Modern и Computer Modern:

Пример текста, сверстанного шрифтами New Computer Modern размера 5pt.
Гарнитура Roman. Гарнитура SansSerif. Гарнитура Typewriter.
Пример текста, сверстанного шрифтами Computer Modern размера 5pt.
Гарнитура Roman. Гарнитура SansSerif. Гарнитура Typewriter.

Гарнитуры EBGaramond, Lato и Anonymous Pro

Гарнитура Garamond (рис. 12.2) одна из старейших в истории полиграфии. Она обладает большим разнообразием хорошо проработанных начертаний, послуживших основой разработки многих других шрифтов. В дистрибутивы TEX входит несколько ее вариантов, среди которых наиболее полной является гарнитура EBGaramond, содержащая шрифты контрастности Regular, Medium, Semibold, Bold и ExtraBold. Их автоматически устанавливает пакет ebgaramond, а математическую гарнитуру Garamond Math пакет garamond-math [91].

[1] Пакет fontsetup-nonfree, устанавливаемый в MiKTEX, в дистрибутиве TEXLive отсутствует.

Шрифты Garamond

Рубленые и моноширинные шрифты должны хорошо выделяться из окружающего текста и при этом гармонировать с ним. Они могут быть чуть крупнее или мельче основного шрифта и отличаться контрастностью.

Стоит помнить, что текст, выделенный изменением *начертания* или **контрастности** самоró основного шрифта, обычно смотрится лучше. Его составляющие сбалансированы оптимально, и чтение минимально утомляет глаза.

```
\oldstylenums  ⇝  1234567890,  1234567890,  1234567890.
```

```
{\tiny Пример текста, сверстанного крошечным шрифтом 5pt. Гарнитура EBGaramond. Гарнитура Lato. Гарнитура Anonymous Pro.}
```

Документ выглядит гармонично, если шрифты текста и формул разработаны в едином стиле, как в данном примере. В «строчных» формулах индексы располагаются сбоку от знака суммирования, а в «вынесенных» — над и под ним. Обратите внимание на отличие выражения $\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!}$ от его аналога:

$$\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!} = \lim_{n\to\infty}\left(1 + \frac{1}{n}\right)^n = \lim_{x\to 0}(1+x)^{1/x} = 2.718281828\ldots$$

В некоторых формулах, например $\lim\limits_{x\to 0}(\mathrm{e}^x - 1)/x = 1$, индексы лучше опустить вниз, а для разделения числителей и знаменателей дробей использовать обычный слэш.

В качестве еще одного примера проинтегрируем вектор Пойнтинга по малому интервалу времени δt и объему магнетика:

$$-\int \mathbf{jE}\,\delta t\,dV = \frac{c}{4\pi}\int \nabla(\mathbf{E}\times\mathbf{H}_m)dV\delta t + \frac{1}{4\pi}\int\left(\mathbf{E}\frac{\partial\mathbf{D}}{\partial t} + \mathbf{H}_m\frac{\partial\mathbf{B}}{\partial t}\right)\delta t\,dV =$$
$$= \frac{c}{4\pi}\oint(\mathbf{E}\times\mathbf{H}_m)d\mathbf{S}\,\delta t + \frac{1}{4\pi}\int(\mathbf{E}\delta\mathbf{D} + \mathbf{H}_m\delta\mathbf{B})dV.$$

Для обозначения переменных используются математические алфавиты и греческие буквы:

ABCDEFGHIJKLMNOPQRTSUVWXYZ abcdefghijklmnopqrtsuvwxyz
ΓΔΘΛΞΠΣΦΨΩ αβγδεεζηθϑικλμνξοπρρστυφφχψω
𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ abcdefghijklmnopqrtsuvwxyz
ABCDEFGHIJKLMNOPQRTSUVWXYZ
abcdefghijklmnopqrtsuvwxyz
𝒜ℬ𝒞𝒟ℰℱ𝒢ℋℐ𝒥𝒦ℒℳ𝒩𝒪𝒫𝒬ℛ𝒯𝒮𝒰𝒱𝒲𝒳𝒴ℤ

Рис. 12.2. Образец шрифтов Garamond, Lato и Anonymous Pro

Все гарнитуры разработаны в единственном размере. Текстовые шрифты поддерживают латиницу, кириллицу и греческий алфавит. Глифы математической гарнитуры имеют контрастность Regular и Bold.

Для подключения шрифтов следует использовать пакет ebgaramond. Он автоматически подгрузит пакет fontspec и настроит текстовые шрифты, а математические нужно загрузить самостоятельно:

```
\usepackage[semibold]{ebgaramond}
\usepackage[scale=0.85, defaultsans]{lato}
\setmonofont{Anonymous Pro}[Scale=0.85]
\usepackage{unicode-math}
\setmathfont{Garamond Math}[Scale=MatchUppercase]
\setmathfont{Garamond Math}%
[range={it/greek->up,bfit/greek->bfup}]
```

Представленная загрузка активирует шрифты контрастности Regular для основного шрифта и Semibold для жирного, а также подключает шрифты Lato в качестве гарнитуры SansSerif и шрифты Anonymous Pro в качестве гарнитуры `Mono`. Образец текста и формул, сверстанный с использованием указанных гарнитур, показан на рис. 12.2.

Заглавные и строчные буквы гарнитур Garamond довольно сильно различаются по высоте, при этом строчные буквы меньше, чем в других шрифтах. Вследствие этого выравнивание других гарнитур методами MatchLowercase или MatchUppercase дает либо слишком мелкие, либо слишком крупные шрифты. В представленной загрузке строчные буквы гарнитуры Garamond увеличены до букв шрифтов Computer Modern, для шрифтов Lato подобран подходящий масштабирующий коэффициент, а к шрифтам Anonymous Pro, имеющим буквы большой ширины, применен метод выравнивания MatchLowercase. Математическая гарнитура Garamond Math выравнивается методом MatchUppercase.

К сожалению, у пакета ebgaramond нет документации, поэтому перечислим параметры его загрузки, отметив прежде, что по умолчанию основной и математический шрифты имеют контрастность Regular, а жирный — Bold, цифры «естественной» ширины печатаются в старом стиле:

scale | scaled — масштабирование шрифтов;

m | medium | sb | semibold | eb | extrabold — активация шрифтов контрастности Medium, Semibold и ExtraBold;

lf | linin | nf — использование обычного стиля цифр;

oldstyle | osf — использование цифр старого стиля;

p | proportional — использование цифр «естественной» ширины;

t | tabular — использование цифр одинаковой ширины.

Пакет ebgaramond вводит ряд команд для печати чисел в разных форматах. Команды `\liningnums{`*числа*`}` и `\oldstylenums{`*числа*`}` выводят их в обычном и старом стиле, а команды `\tabularnums{`*числа*`}` и `\proportionalnums{`*числа*`}` печатают цифры одинаковой и разной ширины.

Команды `\textsu{`*текст*`}` и `\textinf{`*текст*`}` верстают *текст* в виде верхних и нижних индексов. Для печати используются особые глифы, имеющиеся только для латинских букв и цифр, поэтому другие символы *текста* остаются в строке.

Пакет ebgaramond содержит шрифт буквиц,[2] работа над которыми еще не завершена, и их комплект далеко не полон:

Для печати буквиц определена декларация `\initials` и команда `\textin{`*буква*`}`.

В качестве гарнитуры SansSerif к шрифтам Garamond подходят шрифты Lato и Calibri, а гарнитурой Mono может служить шрифт Anonymous Pro. Все они разработаны в единственном размере.

Гарнитуру Anonymous Pro можно *установить*, загрузив пакет anonymouspro, но *подключать* ее следует, как описано в данной главе. Ее шрифты не требуют иных настроек, кроме масштабирования.

Использованные в обсуждавшейся ранее загрузке шрифты Lato устанавливает, подключает и настраивает пакет lato [92]. Перечислим его настройки:

scale | scaled — масштабирование шрифтов;

default | defaultsans — подключение Lato в качестве основного шрифта или гарнитуры SansSerif;

[2] Бу́квица — крупная, отличная от прочих, первая буква главы, раздела или целой книги.

linin | oldstyle — использование обычного или старого стиля печати цифр;

proportional | tabular — печать цифр «естественной» или одинаковой ширины.

Параметры, используемые по умолчанию, выделены жирным шрифтом.

Пакет lato не определяет собственных команд, но команды пакета ebgaramond, введенные для печати цифр и индексов, действуют и на его шрифты.

Гарнитуры STIX2, Lato и Anonymous Pro

Ведущие мировые издательства спонсируют создание общедоступных шрифтов STIX (Scientific and Technical Information eXchange), предназначенных для верстки научных и технических документов. Изначально проект был направлен на разработку математических шрифтов, но начиная с версии STIX2 к ним добавлены и текстовые. В настоящее время математическая гарнитура STIX2 насчитывает 2422 символа [88], а текстовые шрифты поддерживают все европейские алфавиты. Шрифты STIX разрабатываются в единственном размере. Их актуальные версии доступны на сайте `https://github.com/stipub/stixfonts`.

Гарнитуры STIX2 нужно устанавливать вручную, используя пакет stix2-otf, а подключать их удобно с помощью пакета fontsetup:

```
\usepackage[stixtwo]{fontsetup}
\usepackage[scale=0.95, defaultsans]{lato}
\setmonofont{Anonymous Pro}
\setmathfont{STIX Two Math}[range={it/greek->up,bfit/greek->bfup}]
```

При такой загрузке для верстки формул автоматически подключается гарнитура STIX Two Math, поэтому дополнительно объявлять ее не требуется, и в список команд добавлена лишь декларация `\setmathfont`, заменяющая курсивные греческие буквы прямыми.

С гарнитурами STIX2 хорошо сочетаются шрифты Lato и Anonymous Pro, масштабируемые методом MatchUppercase. Образец текста и формул, сверстанный с использованием данных шрифтов, показан на рис. 12.3.

Гарнитуры Times New Roman, Calibri, Courier New и XITS

Версии v.2.82 высококачественных гарнитур Times New Roman и Courier New © Monotype Corporation доступны во многих операционных системах, так как входят в коллекцию “Core fonts for the Web”,

Шрифты STIX2

Рубленые и моноширинные шрифты должны хорошо выделяться из окружающего текста и при этом гармонировать с ним. Они могут быть чуть крупнее или мельче основного шрифта и отличаться контрастностью.

Стоит помнить, что текст, выделенный изменением *начертания* или **контрастности** самого́ основного шрифта, обычно смотрится лучше. Его составляющие сбалансированы оптимально, и чтение минимально утомляет глаза.

```
\oldstylenums  ⇝  1234567890,  1234567890,  1234567890.
```

```
{\tiny Пример текста, сверстанного крошечным шрифтом 5pt. Гарнитура STIX Two Text. Гарнитура Lato. Гарнитура Anonymous Pro.}
```

Документ выглядит гармонично, если шрифты текста и формул разработаны в едином стиле, как в данном примере. В «строчных» формулах индексы располагаются сбоку от знака суммирования, а в «вынесенных» — над и под ним. Обратите внимание на отличие выражения $\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!}$ от его аналога:

$$\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = \lim_{x\to 0}(1+x)^{1/x} = 2.718281828\ldots$$

В некоторых формулах, например $\lim\limits_{x\to 0}(\mathrm{e}^x - 1)/x = 1$, индексы лучше опустить вниз, а для разделения числителей и знаменателей дробей использовать обычный слэш.

В качестве еще одного примера проинтегрируем вектор Пойнтинга по малому интервалу времени δt и объему магнетика:

$$-\int \mathbf{jE}\,\delta t\,dV = \frac{c}{4\pi}\int \nabla(\mathbf{E}\times\mathbf{H}_m)dV\delta t + \frac{1}{4\pi}\int\left(\mathbf{E}\frac{\partial\mathbf{D}}{\partial t} + \mathbf{H}_m\frac{\partial\mathbf{B}}{\partial t}\right)\delta t\,dV = \\ = \frac{c}{4\pi}\oint(\mathbf{E}\times\mathbf{H}_m)d\mathbf{S}\,\delta t + \frac{1}{4\pi}\int(\mathbf{E}\delta\mathbf{D} + \mathbf{H}_m\delta\mathbf{B})dV.$$

Для обозначения переменных используются математические алфавиты и греческие буквы:

ABCDEFGHIJKLMNOPQRTSUVWXYZ abcdefghijklmnopqrtsuvwxyz

ΓΔΘΛΞΠΣΦΨΩ αβγδεεζηθϑικλμνξοπρϱστυφφχψω

𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ 𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷

𝔸𝔹ℂ𝔻𝔼𝔽𝔾ℍ𝕀𝕁𝕂𝕃𝕄ℕ𝕆ℙℚℝ𝕊𝕋𝕌𝕍𝕎𝕏𝕐ℤ 𝕒𝕓𝕔𝕕𝕖𝕗𝕘𝕙𝕚𝕛𝕜𝕝𝕞𝕟𝕠𝕡𝕢𝕣𝕤𝕥𝕦𝕧𝕨𝕩𝕪𝕫

𝒜ℬ𝒞𝒟ℰℱ𝒢ℋℐ𝒥𝒦ℒℳ𝒩𝒪𝒫𝒬ℛ𝒮𝒯𝒰𝒱𝒲𝒳𝒴𝒵 𝒶𝒷𝒸𝒹ℯ𝒻ℊ𝒽𝒾𝒿𝓀𝓁𝓂𝓃ℴ𝓅𝓆𝓇𝓈𝓉𝓊𝓋𝓌𝓍𝓎𝓏

Рис. 12.3. Образец шрифтов STIX2, Lato и Anonymous Pro

Шрифты Times New Roman и XITS

Рубленые и моноширинные шрифты должны хорошо выделяться из окружающего текста и при этом гармонировать с ним. Они могут быть чуть крупнее или `мельче основного шрифта` и отличаться контрастностью.

Стоит помнить, что текст, выделенный изменением *начертания* или **контрастности** самого́ основного шрифта, обычно смотрится лучше. Его составляющие сбалансированы оптимально, и чтение минимально утомляет глаза.

```
\oldstylenums ⇝ 1234567890, 1234567890, 1234567890.
```

```
{\tiny Пример текста, сверстанного крошечным шрифтом 5pt. Гарнитура Times New Roman. Гарнитура Calibri. Гарнитура Courier New.}
```

Документ выглядит гармонично, если шрифты текста и формул разработаны в едином стиле, как в данном примере. В «строчных» формулах индексы располагаются сбоку от знака суммирования, а в «вынесенных» — над и под ним. Обратите внимание на отличие выражения $\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!}$ от его аналога:

$$\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = \lim_{x\to 0}(1+x)^{1/x} = 2.718281828\ldots$$

В некоторых формулах, например $\lim\limits_{x\to 0}(\mathrm{e}^x - 1)/x = 1$, индексы лучше опустить вниз, а для разделения числителей и знаменателей дробей использовать обычный слэш.

В качестве еще одного примера проинтегрируем вектор Пойнтинга по малому интервалу времени δt и объему магнетика:

$$-\int \mathbf{j}\mathbf{E}\,\delta t\,dV = \frac{c}{4\pi}\int \nabla(\mathbf{E}\times\mathbf{H}_m)dV\delta t + \frac{1}{4\pi}\int\left(\mathbf{E}\frac{\partial\mathbf{D}}{\partial t} + \mathbf{H}_m\frac{\partial\mathbf{B}}{\partial t}\right)\delta t\,dV = $$
$$= \frac{c}{4\pi}\oint(\mathbf{E}\times\mathbf{H}_m)d\mathbf{S}\,\delta t + \frac{1}{4\pi}\int(\mathbf{E}\delta\mathbf{D} + \mathbf{H}_m\delta\mathbf{B})dV.$$

Для обозначения переменных используются математические алфавиты и греческие буквы:

$ABCDEFGHIJKLMNOPQRTSUVWXYZ$

$abcdefghijklmnopqrtsuvwxyz$

ΓΔΘΛΞΠΣΦΨΩ αβγδεεζηθϑικλμνξοπρϱστυφφχψω

𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ

𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷

𝖠𝖡𝖢𝖣𝖤𝖥𝖦𝖧𝖨𝖩𝖪𝖫𝖬𝖭𝖮𝖯𝖰𝖱𝖲𝖳𝖴𝖵𝖶𝖷𝖸𝖹 𝖺𝖻𝖼𝖽𝖾𝖿𝗀𝗁𝗂𝗃𝗄𝗅𝗆𝗇𝗈𝗉𝗊𝗋𝗌𝗍𝗎𝗏𝗐𝗑𝗒𝗓

$\mathcal{ABCDEFGHIJKLMNOPQRTSUVWXYZ}$

Рис. 12.4. Образец шрифтов Times New Roman, Calibri, Courier New и XITS

выпущенную в 1996 г. корпорацией Microsoft для свободного использования. Лицензионные ограничения не позволяет распространять более поздние версии шрифтов, поэтому они отсутствуют в дистрибутивах TEX. В то же время действующие лицензии предоставляют право их некоммерческого использования в печати и электронных документах.

Перечисленные шрифты автоматически устанавливаются в ОС Windows. Для установки в других операционных системах[3] шрифты версии v.2.82 доступны на сайте
`https://sourceforge.net/projects/corefonts/files/thefonts/final`.

Отсутствующую в свободном доступе гарнитуру Calibri © Microsoft Corporation можно заменить гарнитурой Lato.

Гарнитура Times New Roman хорошо сочетается с математическими шрифтами STIX2 и XITS. Гарнитура XITS Math — поддерживаемая в настоящее время модификация первой версии шрифтов STIX, содержит 2427 символов [88]. В дистрибутивах TEX ее устанавливает одноименный пакет xits, а актуальные версии шрифтов имеются в свободном доступе на сайте разработчика: `https://github.com/aliftype/xits`.

Подключать шрифты лучше самостоятельно:

```
\usepackage{unicode-math}
\setmainfont{Times New Roman}
\setsansfont{Calibri}
\setmonofont{Courier New}
\setmathfont{XITS Math}
\setmathfont{XITS Math}[range={it/greek->up,bfit/greek->bfup}]
```

Все гарнитуры хорошо сочетаются друг с другом и выравнивать их не стоит. Сверстанный ими образец текста и формул показан на рис. 12.4.

Гарнитуры XITS, Lato и AnonymousPro

Свободно распространяемая гарнитура XITS является наиболее адекватной заменой шрифтов Times New Roman. Ее устанавливает пакет xits вместе с математическими шрифтами. Хотя гарнитура не богата шрифтами различных начертаний, в ней есть прямые и курсивные шрифты, используемые в подавляющем большинстве случаев. Как показано на рис. 12.5, шрифты прекрасно сочетаются с гарнитурами

[3] В ОС Linux коллекцию “Core fonts for the Web” обычно устанавливают пакеты, имя которых содержит слово `mscorefonts` или `msttcorefonts`. Если такого пакета нет, загрузите архивы (файлы расширением *.exe) с указанного сайта, используя программу cabextact, извлеките шрифты и установите их в операционной системе.

Шрифты XITS, Lato и AnonymousPro

Рубленые и моноширинные шрифты должны хорошо выделяться из окружающего текста и при этом гармонировать с ним. Они могут быть чуть крупнее или мельче основного шрифта и отличаться контрастностью.

Стоит помнить, что текст, выделенный изменением *начертания* или **контрастности** самого основного шрифта, обычно смотрится лучше. Его составляющие сбалансированы оптимально, и чтение минимально утомляет глаза.

```
\oldstylenums ⇝ 1234567890, 1234567890, 1234567890.
{\tiny Пример текста, сверстанного крошечным шрифтом 5pt. Гарнитура XITS. Гарнитура Lato. Гарнитура AnonymousPro.}
```

Документ выглядит гармонично, если шрифты текста и формул разработаны в едином стиле, как в данном примере. В «строчных» формулах индексы располагаются сбоку от знака суммирования, а в «вынесенных» — над и под ним. Обратите внимание на отличие выражения $\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!}$ от его аналога:

$$\mathrm{e} = 1 + \sum_{k=1}^{\infty} \frac{1}{k!} = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n = \lim_{x\to 0}(1+x)^{1/x} = 2.718281828\ldots$$

В некоторых формулах, например $\lim\limits_{x\to 0}(\mathrm{e}^x - 1)/x = 1$, индексы лучше опустить вниз, а для разделения числителей и знаменателей дробей использовать обычный слэш.

В качестве еще одного примера проинтегрируем вектор Пойнтинга по малому интервалу времени δt и объему магнетика:

$$-\int \mathbf{jE}\,\delta t\,dV = \frac{c}{4\pi}\int \nabla(\mathbf{E}\times\mathbf{H}_m)dV\delta t + \frac{1}{4\pi}\int \left(\mathbf{E}\frac{\partial\mathbf{D}}{\partial t} + \mathbf{H}_m\frac{\partial\mathbf{B}}{\partial t}\right)\delta t\,dV = $$
$$= \frac{c}{4\pi}\oint (\mathbf{E}\times\mathbf{H}_m)d\mathbf{S}\,\delta t + \frac{1}{4\pi}\int (\mathbf{E}\delta\mathbf{D} + \mathbf{H}_m\delta\mathbf{B})dV.$$

Для обозначения переменных используются математические алфавиты и греческие буквы:

ABCDEFGHIJKLMNOPQRTSUVWXYZ
abcdefghijklmnopqrtsuvwxyz
ΓΔΘΛΞΠΣΦΨΩ αβγδεεζηθϑικλμνξοπρϱστυφφχψω
𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔘𝔙𝔚𝔛𝔜ℨ
𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷
𝔸𝔹ℂ𝔻𝔼𝔽𝔾ℍ𝕀𝕁𝕂𝕃𝕄ℕ𝕆ℙℚℝ𝕊𝕋𝕌𝕍𝕎𝕏𝕐ℤ 𝕒𝕓𝕔𝕕𝕖𝕗𝕘𝕙𝕚𝕛𝕜𝕝𝕞𝕟𝕠𝕡𝕢𝕣𝕤𝕥𝕦𝕧𝕨𝕩𝕪𝕫
𝒜ℬ𝒞𝒟ℰℱ𝒢ℋℐ𝒥𝒦ℒℳ𝒩𝒪𝒫𝒬ℛ𝒮𝒯𝒰𝒱𝒲𝒳𝒴𝒵

Рис. 12.5. Образец шрифтов XITS, Lato и AnonymousPro

Lato и AnonymousPro. Подключать их надо вручную, соблюдая следующий порядок:

```
\usepackage{unicode-math}
\setmainfont{XITS}
\setsansfont{lato}[Scale=0.92]
\setmonofont{Anonymous Pro}[Scale=MatchLowercase]
\setmathfont{XITS Math}
\setmathfont{XITS Math}[range={it/greek->up,bfit/greek->bfup}].
```

12.1.5. Заключительные замечания

Описанным в предыдущем разделе способом, при необходимости дополненным выравниванием шрифтов, можно подключить любой набор OpenType гарнитур и получить доступ к шрифтам основных начертаний и AnonymousPro контрастностей. В подавляющем большинстве случаев этого вполне достаточно для качественной верстки. В то же время некоторые шрифты, например вычурные, останутся за пределами *автоматического* подключения, их нужно активировать вручную. В конце главы рассмотрены богатые возможности настройки шрифтов, имеющиеся в арсенале пакетов fontspec и unicode-math, которые открывают доступ ко всем ресурсам гарнитур и позволяют составлять шрифты, комбинируя ресурсы разных гарнитур. Чтобы понять, как это делать, мы сначала обсудим оперирование шрифтами на уровне ядра LaTeX.

12.2. Работа со шрифтами в схеме NFSS

Связь шрифтов LaTeX с печатными шрифтами обеспечивает механизм, называемый схемой выбора шрифта NFSS (new font selection scheme). Он с успехом работает в LaTeX уже более трех десятков лет, обладает большой гибкостью и эффективностью. Чтобы описать его действие, рассмотрим представление гарнитур на уровне ядра LaTeX.

Ранее говорилось, что PdfLaTeX оперирует 8-битными шрифтами. Описания их гарнитур хранятся в файлах с расширением •.fd. Имена файлов составляются из обозначения кодировки и названия гарнитуры, например t2a•.fd для кодировки T2A, поэтому список файлов позволяет легко определить список кодировок, поддерживаемых гарнитурой.

Описания формируют декларации

```
\DeclareFontFamily{кодировка}{семейство}{спец. символ} ,
\DeclareFontShape{код.}{сем.}{контр.}{нач.}{шрифт}{} .
```

Первая вводит семейство шрифтов (гарнитуру) определенной кодировки. В ее последнем аргументе указывают символы, служащие для специальных целей. В текстовых шрифтах это может быть знак переноса, на котором можно разбивать строки, а в математических — символ, служащий для тонкой настройки акцентов. Часто последний аргумент остается пустым.

Вторая декларация вводит шрифт с определенным набором атрибутов. Пара ее первых аргументов аналогична аргументам декларации `\DeclareFontFamily`. Два следующих задают контрастность и начертание шрифта, а в пятом указывается печатный шрифт, для которого перечисляются имеющиеся размеры и методы получения других размеров [4].

Последний аргумент, как правило, остающийся пустым, позволяет корректировать параметры шрифта, заданные в его метриках, такие как величины длин em и ex, наклон символов, ширина межсловного интервала и т. д. [4]. Метрики шрифтов хранятся в файлах с расширением •.tfm. В них описаны взаимные связи и размеры символов, представленные в виде параметров боксов и расстояний между ними. Именно боксы используются в верстке вместо самих символов.

Предваряя обсуждение того, как атрибуты шрифтов настраиваются на уровне команд, адресующихся к ядру LaTeX, опишем общую процедуру их изменения. Она состоит из двух действий: сначала значение атрибута заявляется, а затем его активируют декларацией `\selectfont`. Например, `\fontencoding{T1}\selectfont`. Значения различных атрибутов задают различные команды.

Выполняя активацию, компилятор проверяет корректность значения и, если оно определено, изменяет шрифт. В противном случае атрибут и шрифт остаются неизменными. Если значение заявляется, но не активируется сразу, шрифт не меняется, однако новое значение может вступить в силу впоследствии, когда будут меняться другие атрибуты. Их можно менять сразу несколько. Для этого нужно перечислить команды, задающие новые значения, и применить общую декларацию `\selectfont`, поставив ее в конце. Она выполнит проверку и активацию всех объявленных ранее замен.

12.2.1. Активация кодировок 8-битных шрифтов

Как уже говорилось в разд 2.6.4, при верстке русского текста автоматически загружаются кодировки T1, TS1 и T2A, и последняя активируется. Дополнительные кодировки подключают с помощью пакета fontenc:

\usepackage[*список кодировок*]{fontenc}.

Кодировки перечисляются списком через запятую в параметре загружающей его команды. Последняя из них становится активной. Ее символы используются «непосредственно», а символы других кодировок нужно вводить командами, или объявлять дополнительно, как описано в прил. Б.1

Кодировку, используемую по умолчанию, в начале верстки активирует набор команд \fontencoding\encodingdefault\selectfont. Заявляют и активируют ее команда \fontencoding{*кодировка*} и декларация \selectfont. Наименование кодировки хранит команда \encodingdefault. В начале верстки ядро LaTeX присваивает ей значение T1, а пакет babel заменяет его значением T2A. Присваивание выполняют декларации \newcommand или \renewcommand, например \renewcommand\encodingdefault{T2A}.

Активируя кодировку XX, компилятор проверяет, загружена ли она, и есть ли среди описаний гарнитур загруженных шрифтов файлы, содержащие декларации

\DeclareFontFamily{XX}{•}{•},
\DeclareFontShape{XX}{•}{•}{•}{•}{•}.

Если проверка дала положительный результат, компилятор начнет верстать документ шрифтами данной кодировки.

12.2.2. Выбор гарнитур

Набор команд, приведенных ранее в табл. 2.6, обеспечивает управление гарнитурами на уровне пользователя. Декларации \rmfamily, \sffamily и \ttfamily вместе с командами \textrm{•} и т. п. активируют шрифты трех стандартных гарнитур. Команды организованы с помощью этих деклараций, например \textrm{•} = {\rmfamily •}, поэтому мы обсудим лишь устройство последних:

\rmfamily = \fontfamily\rmdefault\selectfont,
\sffamily = \fontfamily\sfdefault\selectfont,
\ttfamily = \fontfamily\ttdefault\selectfont.

Команды `\fontfamily{`*гарнитура*`}` заявляют идентификаторы гарнитур, которые хранят команды `\rmdefault`, `\sfdefault` и `\ttdefault`. Для шрифтов Computer Modern они имеют следующие значения: `\rmdefault` ≡ `cmr`, `\sfdefault` ≡ `cmss` и `\ttdefault` ≡ `cmtt`. Используя переопределения

```
\renewcommand\rmdefault{...},
\renewcommand\sfdefault{...},
\renewcommand\ttdefault{...},
```

издатели легко адаптируют шрифтовое оформление документа к своим требованиям. Разместив данные команды в преамбуле рукописи, можно настроить стандартные гарнитуры по своему усмотрению.

В команду `\familydefault` помещается идентификатор гарнитуры основного шрифта, активируемого в начале верстки декларациями

```
\fontfamily\familydefault\selectfont.
```

Как правило, он соответствует гарнитуре Serif, но если изменить его декларацией `\renewcommand`, например

```
\renewcommand\familydefault\sfdefault,
```

основным станет рубленый шрифт. Последовательность команд

`\fontfamily{`*гарнитура*`}\selectfont`

позволяет использовать любую гарнитуру, если ее идентификатор введен в загруженных описаниях гарнитур декларациями

```
\DeclareFontFamily{•}{гарнитура}{•},
\DeclareFontShape{•}{гарнитура}{•}{•}{•}{•}.
```

Гарнитуру можно менять в любом месте документа. Гарнитура, измененная внутри группы, восстанавливается по выходе из нее. Декларации `\rmfamily`, `\sffamily` и `\ttfamily` позволяют вернуться к стандартной гарнитуре, когда действие нестандартной не было ограничено группой.

Приведем пример подключения гарнитуры SansMono коллекции шрифтов DejaVu: `\fontfamily{DejaVuSansMono-TLF}\selectfont`. Идентификаторы современных гарнитур чаще всего составляют из названия печатного шрифта и стиля вывода чисел, например Garamond-Osf или LibertinusSerif-TLF. Аббревиатуры LF и OsF обозначают числа обычного и старого стиля. Если числа представляются в формате, в котором цифрам отводятся позиции одинаковой ширины, к ним добавляется префикс T. Таким образом, TLF означает обычный стиль чисел моноширинного формата.

12.2.3. Управление контрастностью

Хорошо проработанные шрифты имеют большой диапазон контрастности, определяемой сочетанием ширины и насыщенности глифов. В системе NFSS они обозначаются аббревиатурами, перечисленными в табл. 12.2.

Таблица 12.2

Ширина и насыщенность шрифтов

Ширина глифов, %			Насыщенность	
uc	50	Ultra Condensed	ul	Ultra Light
ec	62	Extra Condensed	el	Extra Light
c	75	Condensed	l	Light
sc	87.5	Semi Condensed	sl	Semi Light
m	100	Medium	m	Medium
sx	112.5	Semi Expanded	sb	Semi Bold
x	125	Expanded	b	Bold
ex	150	Extra Expanded	eb	Extra Bold
ux	200	Ultra Expanded	ub	Ultra Bold

Таблица 12.3

Аббревиатуры контрастности шрифтов

	ul	el	l	sl	sb	b	eb	ub
ec	ulec	elec	lec	slec	sbec	bec	ebec	ubec
c	ulc	elc	lc	slc	sbc	bc	ebc	ubc
sc	ulsc	elsc	lsc	slsc	sbsc	bsc	ebsc	ubsc
x	ulx	elx	lx	slx	sbx	bx	ebx	ubx

Обозначение контрастности определяет комбинация ширины и насыщенности. Контрастность шрифтов обычной ширины (Medium) обозначается аббревиатурой насыщенности, например `sb` для полужирного шрифта. Контрастность шрифтов обычной насыщенности (также Medium) обозначается аббревиатурой ширины глифов, например `uc` для сверхузкого шрифта. Мнемоника остальных обозначений контрастности, приведенных в табл. 12.3, строится по простому принципу: к аббревиатуре насыщенности добавляется аббревиатура ширины, например `slc` для полусветлого (`sl`) узкого (`c`) шрифта. Сверхузкие (`uc`), полуширокие (`sx`) и сверхширокие (`ex` и `ux`) шрифты обычно имеют лишь одну насыщенность, поэтому их контрастность обозначается так же, как ширина.

В стандартных гарнитурах Computer Modern, имеющих лишь две контрастности, на уровне пользователя определены только шрифт средней ширины обычной насыщенности, активируемый декларацией `\mdseries`, и широкий шрифт жирной насыщенности, активируемый декларацией `\bfseries` (см. табл.2.6).[4] Данные декларации введены как

```
\bfseries = \fontfseries\bfdefault\selectfont,
\mdseries = \fontfseries\mddefault\selectfont,
```

где команды `\fontfseries{`*контрастность*`}` заявляют *контрастности* обычных (`m`) и жирных (`bx`) шрифтов, аббревиатуры которых хранят команды `\mddefault` и `\bfdefault`. В скобках указаны их стандартные значения. В команду `\seriesdefault` записывается обозначение контрастности, используемой по умолчанию. Обычно оно устанавливается равным значению `\mddefault`.

В целях унификации, описания шрифтов Computer Modern делают контрастности `bx` и `b` синонимами. Это позволяет декларации `\bfseries` активировать жирный шрифт гарнитур разной контрастности, например `b` для Mono и `bx` для Serif и SansSerif.

Последовательность команд `\fontfseries{•}\selectfont` позволяет подключить шрифт любой стандартной контрастности из набора NFSS, приведенного в табл. 12.2 и 12.3. Декларация

`\DeclareFontShape{•}{•}{`*обозначение контрастности*`}{•}{•}{•}`

дает возможность ввести обозначение для нестандартной контрастности непосредственно в описании гарнитуры. Если загруженное описание содержит такую декларацию, нужный шрифт подключит набор команд `\fontseriesforce{`*контрастность*`}\selectfont`.

Для активации часто используемой контрастности, отличной от `m`, `b` или `bx`, можно ввести декларацию, аналогичную стандартным. Например, хорошо проработанные гарнитуры рубленых шрифтов содержат светлые шрифты контрастности `l`. К сожалению, LaTeX пока еще не имеет удобного интерфейса работы с ними на уровне пользователя. Определим для их подключения декларацию `\lfseries`:

```
\newcommand\lfseries{\fontfseries{lc}\selectfont},
```

и воспользуемся ей:[5] `{\Lato\lfseries` пример светлого шрифта `}`. Используя ее, следует помнить, что компиляция текста, в котором она при-

[4] Строго говоря, в гарнитуре `cmr` есть прямой шрифт насыщенности `b`, а в гарнитуре `cmss` — прямой шрифт насыщенности `sbc`, но обычно они не используются.

[5] Декларация `\Lato`, подключающая одноименную гарнитуру, определена на с. 322.

сутствует, скопированного в документ, в котором она не определена, вызовет ошибку. В то же время попытки активации шрифта командами `\fontfseries` и `\fontseriesforce` в том же тексте будут просто игнорироваться.

12.2.4. Изменение начертаний

Шрифты обладают большим разнообразием начертаний. Помимо традиционных прямых, наклонных и курсивных шрифтов гарнитуры могут содержать вычурные шрифты и капитель, а также их комбинации с остальными начертаниями. Типы начертаний, аббревиатуры, используемые в описаниях гарнитур, а также активирующие их декларации приведены в табл. 12.4. Прямой курсив (`ui`), имеющийся в гарнитуре Computer Modern Roman, в других шрифтах встречается очень редко.

Таблица 12.4

Начертания шрифтов

Начертания		Декларации
`n`	Основное начертание	`\normalshape`
`up`	Прямой шрифт	`\upshape`
`it`	*Курсив*	`\itshape`
`ui`	Прямой курсив	`\fontshape{ui}\selectfont`
`sl`	*Наклонный шрифт*	`\slshape`
`sw`	*Вычурный шрифт*	`\swshape`
`sc`	Капитель	`\scshape`
`ssc`	Разреженная капитель	`\sscshape`
`scit`	*Курсивная капитель*	`\scshape\itshape`
`scsl`	*Наклонная капитель*	`\scshape\slshape`
`scsw`	*Вычурная капитель*	`\scshape\swshape`
`ulc`	*Не* капитель	`\ulcshape`

Начертание основного шрифта, который обычно бывает прямым, обозначается буквой `n`, но во избежание возможных коллизий для прямого шрифта имеется собственная аббревиатура `up`. Аббревиатура `ulc` и декларация `\ulcshape` не соответствуют какому-либо реальному шрифту. Они используются для того, чтобы восстановить начертание строчных букв, если ранее использовалась капитель. Обрабатывая декларацию `\ulcshape`, компилятор «изымает» буквы `sc` из обозначе-

ния комбинации начертаний, сохраняя при этом наклонное, курсивное или вычурное начертание. Обычной или разреженной капители он присваивает атрибут `up`, т. е. активирует обычный прямой шрифт.

Команды `\updefault`, `\itdefault`, `\sldefault`, `\swdefault` и `\scdefault` хранят аббревиатуры соответствующих начертаний. Их значения, устанавливаемые по умолчанию, указаны в табл. 12.4. Начертание стандартного шрифта хранит команда `\shapedefault`. Декларации, активирующие стандартные начертания, заданы с помощью перечисленных выше команд, например

`\upshape` = `\fontshape\updefault\selectfont` и т. п.

Их можно легко адаптировать под имеющуюся конфигурацию шрифтов. Например, если шрифт содержит прямой вычурный шрифт, обозначенный как `nw`, чтобы активировать его декларацией `\swshape`, достаточно сделать переопределение `\renewcommand\swdefault{nw}`.

Команды `\fontshape{`*начертание*`}\selectfont` позволяют оперировать любыми начертаниями шрифтов, объявленными декларациями `\DeclareFontShape` в описании гарнитуры. Например, среди перечисленных в табл. 12.4 стандартных начертаний отсутствуют рукописные шрифты. Декларацией `\DeclareFontShape{•}{•}{•}{scr}{•}{}` им можно задать обозначение `scr` и подключать стандартным образом: `\fontshape{scr}\selectfont`.

12.2.5. Активация шрифта с заданными атрибутами

Для подключения шрифта с заданным набором всех атрибутов, кроме кегля, предусмотрена декларация

`\usefont{`*кодировка*`}{`*гарнитура*`}{`*контрастность*`}{`*начертание*`}`,

имеющая такие же аргументы, что и `\DeclareFontShape`. Например, так можно активировать жирный курсив гарнитуры NimbusSerif с буквами латиницы: `\usefont{T1}{NimbusSerif}{bx}{it}`. Определение команды `\usefont` содержит декларацию `\selectfont`, поэтому она самостоятельно установит все атрибуты.

12.2.6. Резюме

Подведем промежуточный итог. Подключая шрифт, компилятор по названию кодировки и гарнитуры находит файл с ее описанием. Из команд `\DeclareFontShape` он определяет обозначение атрибутов

и сравнивает их с запрашиваемыми. Если шрифт с требуемыми атрибутами найден, он активируется, а в противном случае запрос на подключение шрифта отклоняется.

Система NFSS сочетает в себе простоту устройства с возможностью оперировать большим числом атрибутов шрифтов, используя минимум средств. Например, внутренняя организация деклараций, устанавливающих основной шрифт документа и подключающих произвольный шрифт, отличается лишь значениями атрибутов:

```
\normalfont = \fontencoding\encodingdefault
              \fontfamily\familydefault  \fonseries\seriesdefault
              \fontshape\shapedefault    \selectfont,

\usefont}{T2A}{EBGaramond-LF}{b}{nw} =   \fontencoding{T2A}
              \fontfamily{EBGaramond-LF} \fontseries{b}
              \fontshape{nw}             \selectfont.
```

Наиболее полно и последовательно система NFSS описана к книге [4], в которой заинтересованный читатель получит информацию «из первых рук», так один из ее авторов Франк Миттельбах является также и одним из основоположников NFSS.

12.2.7. Выбор кегля

Каждый кегль высококачественных гарнитур имеет свой шрифт. Глифы шрифтов разных размеров не пропорциональны друг другу. С уменьшением кегля их высота уменьшается быстрее, чем ширина, поэтому шрифты мелких размеров более широкие, а больших размеров более узкие, вдобавок к этому с уменьшением размера глифы делаются жирнее. При пропорциональном уменьшении мелкий текст становится нечитаемым, а крупный — слишком контрастным. Это иллюстрирует на с. 301 сравнение крошечных шрифтов гарнитур Computer Modern и New Computer Modern.

Печатные шрифты разных начертаний и насыщенности даже в одной и той же гарнитуре могут иметь разные наборы кеглей. Перечень имеющихся печатных шрифтов помещается в пятый аргумент команды `\DeclareFontShape`, там же указывается, как они используются для получения шрифтов недостающих размеров. Обычно очень мелкие и очень крупные шрифты генерируются пропорциональным масштабированием из шрифтов минимального и максимального размера, остальные из наиболее близких шрифтов.

На уровне пользователя размер шрифта настраивают декларации, перечисленные ранее в табл. 2.7. Обычно их вполне достаточно для шрифтового оформления документа. Они определены последовательностью команд `\fontsize{`*кегль*`}{`*интервал*`}\selectfont`. Первый аргумент команды `\fontsize` задает размер шрифта, а второй — высоту межстрочного интервала `\baselineskip`. Например, стандартные классы документа для основного шрифта 10pt крошечный вводят как

`\tiny = \fontsize{5pt}{6pt}\selectfont`.

Аргументами команды `\fontsize` являются длины, имеющие значение и размерность, а если используются просто числа, то длины выражаются в пунктах pt. Если шрифт, заявляемый командой `\fontsize{•}{•}`, отсутствует, он автоматически генерируется масштабированием шрифта ближайшего размера.

При изменении кегля иногда требуется дополнительная регулировка межстрочного интервала. Ее выполняет последовательность команд `\linespread{`*число*`}\selectfont`, задающая значение коэффициента `\baselinestretch`, на который умножается длина `\baselineskip` при расчете величины интервала (см. разд. 2.4).

Таким образом, тонкую настройку шрифта можно задать как `\fontsize{•}{•}\linespread{•}\selectfont`, или `\linespread{0.8}\Large`. В последнем случае декларация `\Large`, содержащая в своем определении команду `\selectfont`, активирует как размер шрифта, так и новое значение коэффициента `\baselinestretch`.

При изменении размера шрифта кегль меняется сразу, тогда как межстрочный интервал может остаться неизменным. Дело в том, что он устанавливается для абзаца в целом и, если абзац содержит как крупный, так и мелкий текст, величину интервала определяет его размер в конце абзаца. Разберем пару примеров, в которых части абзацев верстаются разным кеглем:

```
крупный текст ... {\footnotesize мелкий текст ... }\par,
крупный текст ... {\footnotesize мелкий текст ... \par}.
```

Перед декларацией `\footnotesize` шрифт имеет обычный размер, а внутри группирующих скобок он уменьшается. В первом случае межстрочный интервал будет соответствовать обычному тексту, так как абзац формируется командой `\par` *после* закрывающей скобки, когда восстановится начальный размер шрифта. Во втором случае команда `\par` завершает абзац внутри группы, когда активен размер `\footnotesize`, поэтому будет установлен межстрочный интервал мел-

кого текста. При этом расстояние между строками будет изменяться при переходе от крупного текста к мелкому. Для крупного текста оно будет минимизировано так, чтобы строки не накладывались друг на друга, а интервал мелкого текста будет соответствовать размеру `\footnotesize`.

12.2.8. Математические шрифты

Математические шрифты имеют свою специфику. Восьмибитные и OpenType шрифты сильно различаются своим устройством. Не анализируя сложную организацию 8-битных шрифтов, подробно описанную в книге [4], отметим лишь пару общих черт обоих форматов, которые понадобятся нам в дальнейшем.

Математические символы делятся на восемь категорий, определяющих правила их взаимодействия между собой. Коды категорий и команды, причисляющие символы к той или иной категории, приведены в табл. 12.5. Прил. В содержит таблицы символов, разбитых по категориям.

Таблица 12.5

Категории математических символов

Коды	Команды	Категории
0	`\mathord{`*символ*`}`	обычные символы
1	`\mathop{`*символ*`}`	большие операторы и функции
2	`\mathbin{`*символ*`}`	знаки бинарных операций
3	`\mathrel{`*символ*`}`	знаки соотношений
4	`\mathopen{`*символ*`}`	открывающие разделители
5	`\mathclose{`*символ*`}`	закрывающие разделители
6	`\mathpunct{`*символ*`}`	знаки пунктуации
7	`\mathalpha{`*символ*`}`	буквы

Наборы математических шрифтов разного стиля организованы в виде версий. Чтобы изменить стиль верстки формул, нужно *перед* формулой активировать нужную версию декларацией `\mathversion{`*версия*`}`. Шрифты нормальной и жирной насыщенности собраны в версиях normal и bold. У декларации `\mathversion}{bold}`, активирующей жирные шрифты, есть более короткий синоним `\boldmath`. Декларация `\mathversion` является локальной, поэтому верстку жирными шрифтами можно легко ограничить группой.

12.3. Работа с OpenType гарнитурами

Для работы с OpenType гарнитурами компиляторы LuaLaTeX и XeLaTeX используют ресурсы пакета fontspec, который практически полностью совместим с системой NFSS. Начертанием, контрастностью и размером подключаемых шрифтов управляют обсуждавшиеся ранее команды, лишь активация гарнитур отличается от принятой в NFSS, однако и это отличие можно убрать настройками шрифтов. Единственное «глобальное» отличие заключается в том, что у OpenType шрифтов отсутствует атрибут кодировка.

Пакет fontspec [86] обладает хорошо структурированной системой настроек. Настройки шрифтов указываются в командах, часть из которых может находиться непосредственно в рукописи, однако большинство помещается в файлы с описаниями шрифтов и конфигурационный файл fontspec.cfg. Действия самого пакета настраивают параметры его загрузки:

no-config — не загружать конфигурационный файл;

no-math — не настраивать математические шрифты (по умолчанию, чтобы обеспечить единство стиля текста и формул, fonspec заменяет математические алфавиты `\mathrm`, `\mathsf`, `\mathtt` и `\mathbf` шрифтами основных гарнитур Serif, SansSerif и Mono);

quiet — направлять предупреждающие сообщения в •.log файл, не выводя их на консоль;

silent — не выдавать предупреждающие сообщения.

Настройки шрифтов мы обсудим по мере необходимости, а сначала рассмотрим методы подключения гарнитур, предлагаемые пакетом fonspec.

12.3.1. Подключение шрифтов

Как говорилось ранее, основные гарнитуры задают декларации

```
\setmainfont[настройки]{гарнитура}[настройки],
\setsansfont[настройки]{гарнитура}[настройки],
\setmonofont[настройки]{гарнитура}[настройки].
```

Первая устанавливает основной шрифт, которым по умолчанию служит гарнитура Serif, две другие — шрифты гарнитур SansSerif и Mono. Основные гарнитуры не определены жестко, их можно заменить в

любом месте рукописи, повторно используя эти декларации. Наряду с основными в верстке можно задействовать любое количество *дополнительных* гарнитур, введя для их активации собственные декларации:

```
\newfontfamily{декларация}{гарнитура}[настройки],
\renewfontfamily{декларация}{гарнитура}[настройки],
\providefontfamily{декларация}{гарнитура}[настройки],
\setfontfamily{декларация}{гарнитура}[настройки].
```

Первая команда создает декларацию для подключения гарнитуры, вторая заменяет гарнитуру для существующей декларации, третья определяет декларацию, если она не была введена ранее, а последняя делает это, не выполняя никаких проверок, поэтому ей нужно пользоваться с осторожностью.

Создадим для примера декларацию `\Lato`, подключающую одноименную гарнитуру, и применим ее:

```
\providefontfamily\Lato{Lato}[Scale=MatchLowercase]
{\Lato Теперь текст верстается шрифтом Lato, начертание и насыщенность которого изменяют стандартные команды, \textit{курсив}, и декларации \bfseries жирный текст. Чтобы вернуть стандартный шрифт, достаточно выйти из группы }, или использовать одну из деклараций \rmfamily, \sffamily, \ttfamily или \normalfont.
```

Шрифты, составляющие гарнитуру, обычно находятся в разных файлах.[6] В отдельные файлы помещаются прямой и наклонный шрифты, а также курсив и шрифты различной контрастности. Например, показанные на рис. 12.6 шрифты гарнитуры Lato,[7] распределенные в восемнадцати файлах, насчитывают девять контрастностей прямых и курсивных начертаний. Чтобы обратиться к требуемому шрифту, нужно знать устройство гарнитуры.

Гарнитура имеет общее название (Preferred family), а каждый ее шрифт имеет несколько собственных: название в составе гарнитуры (Full name), имя в составе гарнитуры (PostScript name) и имя файла со шрифтом, которое обычно соответствует его имени в составе гарнитуры. Шрифты определенной насыщенности могут иметь несколько начертаний, тогда в рамках гарнитуры они образуют семейство (Family). У

[6] TrueType формат позволяет собрать все шрифты в одном файле, которому дается расширение •.ttc (TrueType Collection).

[7] В левой колонке рисунка указаны обозначения контрастностей и начертаний в системе NFSS.

Гарнитура XLato

ul:n	Artes molliunt mores.	Искусства смягчают нравы.
ul:it	*Artes molliunt mores.*	*Искусства смягчают нравы.*
el:n	Docendo discimus.	Обучая, мы учимся сами..
el:it	*Docendo discimus.*	*Обучая, мы учимся сами.*
l:n	Usus magister est optimus.	Практика лучший учитель.
l:it	*Usus magister est optimus.*	*Практика лучший учитель.*
sl:n	**Errāre humānum est.**	**Человеку свойственно ошибаться.**
sl:it	***Errāre humānum est.***	***Человеку свойственно ошибаться.***
m:n	**Ignorantia non est argumentum.**	**Незнание — не довод.**
m:it	***Ignorantia non est argumentum.***	***Незнание — не довод.***
sb:n	**Aquila non captat muscas.**	**Орел не ловит мух.**
sb:it	***Aquila non captat muscas.***	***Орел не ловит мух.***
b:n	**Asinus asinum fricat.**	**Осел трется об осла.**
b:it	***Asinus asinum fricat.***	***Осел трется об осла.***
eb:n	**Similis simili gaudet.**	**Подобный подобному радуется.**
eb:it	***Similis simili gaudet.***	***Подобный подобному радуется.***
ub:n	**O tempora! O mores!**	**О времена! О нравы!**
ub:it	***O tempora! O mores!***	***О времена! О нравы!***

Рис. 12.6. Пример шрифтов гарнитуры Lato

основного шрифта гарнитуры, как правило это прямой шрифт обычной насыщенности, название семейства соответствует названию гарнитуры. Разберем иерархию шрифтов на примере гарнитуры Lato, для этого воспользуемся информацией о шрифтах, выводимой программой otfinfo : с ключом `i` она печатает полный перечень всех имен. Сравним информацию об основном шрифте и о тонком курсиве:

```
otfinfo -i Lato-Regular.ttf
Family:          Lato
Subfamily:       Regular
Full name:       Lato Regular
PostScript name: Lato-Regular
```

```
otfinfo -i Lato-ThinItalic.ttf
Family:          Lato Thin
Subfamily:       Italic
Full name:       Lato Thin Italic
PostScript name: Lato-ThinItalic
Preferred family:    Lato
Preferred subfamily: Thin Italic
```

Чтобы не дублировать информацию, для основного шрифта выводится только название семейства Lato, это отличает его от других шрифтов,

название семейства которых складывается из названий гарнитуры и насыщенности, как Lato Thin, в нашем случае. Полное имя шрифта содержит информацию о насыщенности (Lato Regular для основного шрифта) и начертании (Lato Thin Italic для тонкого курсива). Имя в составе гарнитуры и имя файла формируются из полного имени шрифта, в котором названия гарнитуры и насыщенности разделяются дефисом, а названия насыщенности и начертания пишутся слитно.

В аргументах обсуждаемых команд *гарнитуру* можно задать как ее собственным названием, так и названием шрифта. При подключении по названию гарнитуры автоматически становятся доступны ее основной шрифт, курсив, капитель и их жирные варианты. В гарнитурах, подключаемых по названию шрифта (обобщим так название семейства, полное имя и имя файла), доступным становится только сам шрифт. Например, при подключении:

```
\newfontfamily\LatoRegular{Lato-Regular.ttf}[Scale=MatchLowercase],
```

в гарнитуре, активируемой декларацией `\LatoRegular`, будет доступен только прямой шрифт, команды, изменяющие контрастность и начертание, на него действовать не будут. Отметим, что в названиях шрифтов и гарнитур заглавные и сточные не различаются, а пробелы можно опустить.

Чтобы подключать шрифты с «фиксированными» атрибутами, fontspec вводит еще один набор деклараций:

```
\newfontface{декларация}{шрифт}[настройки] ,
\renewfontface{декларация}{шрифт}[настройки] ,
\providefontface{декларация}{шрифт}[настройки] ,
\setfontface{декларация}{шрифт}[настройки]
```

Вот как изменится предыдущий пример переписанный с их помощью:

```
\newfontface\LatoRegular{Lato-Regular.ttf}[Scale=MatchLowercase],
{\LatoRegular Теперь текст верстается шрифтом LatoRegular, атрибуты которого,
\bfseries\itshape\small кроме кегля, будут неизменны }.
```

На самом деле команды `\...family` и `\...face` можно рассматривать как синонимы, введенные для классификации подключения шрифтов. При одинаковых аргументах и параметрах они формируют гарнитуры с одинаковым набором атрибутов. Будут ли они неизменны, зависит не от имен команд, а, главным образом, от того, задан ли их второй аргумент названием гарнитуры или названием шрифта. Обрабатывая эти команды, компилятор проверяет наличие шрифта (основного

шрифта, в случае гарнитуры), и если он не найден, выдает сообщение об ошибке и прерывает компиляцию. Доступность шрифтов различных начертаний и контрастностей проверяется уже после активации гарнитуры или шрифта при выполнении команд, изменяющих эти атрибуты. Об отсутствии таких шрифтов компилятор лишь предупреждает.

Шрифты можно проверить заранее, и в зависимости от их наличия выбрать правильный вариант их использования. Это обеспечивает условная конструкция `\IfFontExistsTF{`*шрифт*`}{`*да*`}{`*нет*`}`. Ее первым аргументом является имя файла со шрифтом, которое можно указать без расширения. Если файл находится, выполняется ветвь *да*, в противном случае — ветвь *нет*. В ветвях можно провести дальнейшие проверки и разместить команды, устанавливающие основные и дополнительные гарнитуры, изменяющие настройки шрифтов и т. п.

Возможности поиска шрифтов компиляторов LaTeX и XeTeX различаются. Шрифты операционной системы им доступны как по названиям гарнитур, так и по именам файлов. LaTeX также без ограничений загружает шрифты дистрибутива TeX, а XeTeX находит их только по именам файлов.

12.3.2. Управление контрастностью шрифтов

На примере шрифтов Lato видно, что хорошо проработанные OpenType гарнитуры могут иметь широкий диапазон контрастностей, из которых две основные активируют декларации `\mdseries`, `\bfseries` и команды `\textmd{`•`}`, `\textbf{`•`}`. Доступ к остальным обеспечивают декларации NFSS `\fontseries{`•`}\selectfont`. Для управления контрастностью пакет fontspec строит на их основе собственный набор команд:

`\strongfontdeclare{`*список деклараций*`}`,
`\strong{`*текст*`}`, `\strongreset`.

Аргументом первой команды является *список деклараций* NFSS, изменяющих контрастность. Вторая изменяет контрастность *текста* в соответствии с этим списком. Для этого команды `\strong` вкладываются друг в друга. Уровень вложенности задает контрастность текста в соответствии со списком. Разберем пример:

```
\strongfontdeclare{\fontseries{sb}\selectfont,
        \fontseries{b}\selectfont, \fontseries{eb}\selectfont,
        \fontseries{ub}\selectfont}.
```

Гарнитура Lato содержит жирные шрифты контрастностей `sb, b, eb,`

ub. Если настроим управление жирными шрифтами, задав указанный порядок их изменения, команда \strong{...} будет верстать текст шрифтом sb, вложенная команда подключит шрифт контрастности b, следующие вложения изменят контрастность на eb и ub. Еще более глубокая вложенность не вызовет ошибок, но контрастность текста меняться не будет. Чтобы вернуться к менее контрастному шрифту нужно возвратиться на соответствующий уровень вложенности.

Декларация \strongreset позволяет установить определенный уровень контрастности. По умолчанию он не определен, и декларация ничего не делает. Для примера установим ее на начальный уровень:

\renewcommand\strongreset{\fontseries{sb}\selectfont}.

Теперь декларация \strongreset будет активировать шрифты контрастности sb, находясь в аргументе вложенной команды \strong, она сбросит все уровни вложения, поэтому следующая вложенная команда \strong будет верстать текст шрифтом контрастности eb.

Команды \textmd{•} и \textbf{•}, будучи вложенными в команды \strong, сверстают текст шрифтами контрастности m и b, не изменив порядок вложенности, тогда как декларации \mdseries, \bfseries его нарушат, поэтому в тесте, верстаемом командами \strong, ими лучше не пользоваться.

Наряду с командами \strong пакет fontspec определяет окружение strongenv, действующее аналогичным образом. Хотя команды и окружения выполняют одинаковые действия, смешивать их не стоит. Их вложенность контролируется по отдельности, и использование команд в окружениях или окружений в аргументах команд нарушает порядок следования контрастностей.

12.3.3. Печать чисел

Как правило, OpenType шрифты содержат различные стили цифр. Для печати чисел пакет fontspec добавляет к стандартной команде \oldstylenums{•}, выводящей цифры старого стиля, команду, печатающую обычные цифры: \liningnums{число}. Она требуется, если старый стиль вывода чисел установлен основным.

12.3.4. Настройка OpenType шрифтов

Пакет fontspec использует несколько уровней настройки шрифтов. Вместе с ним загружается конфигурационный файл fontspec.cfg с на-

стройками основных гарнитур Serif, SansSerif и Mono, не привязанными к каким-либо конкретным шрифтам. Затем, как правило, загружаются настройки подключаемых гарнитур, находящиеся либо в файлах пакетов, либо в файлах пользователя, назовем их *описаниями гарнитур*. После этого ряд команд, используемых в рукописи, позволяет корректировать, дополнять или полностью менять загруженные настройки.

Рассмотрим настройку OpenType шрифтов, начав с простейшего примера подключения шрифтов Lato декларацией:

```
\setsansfont{Lato}[Scale = MatchLowercase]
```

При таком подключении будут доступны прямой шрифт и курсив насыщенности Regular и Bold. При комбинировании с контрастными гарнитурами Serif, например STIX Two Text или Times New Roman, контрастность гарнитуры Lato желательно увеличить до Medium, но декларация `\setsansfont{Lato-Medium}` обеспечит только прямой шрифт, так как гарнитура подключается по его названию. В этом случае курсив и жирные шрифты нужно явно прописать в настройках:

```
\setsansfont{Lato-Medium}[ Scale = MatchLowercase,
            UprightFont    = Lato-Medium.ttf,
            BoldFont       = Lato-Bold.ttf,
            ItalicFont     = Lato-MediumItalic.ttf,
            BoldItalicFont = Lato-BoldItalic.ttf ]
```

Расширения имен файлов можно опустить, но если в системе установлены как OpenType, так и TrueType версии шрифтов, их лучше указать явно. С помощью параметра Extension это можно сделать сразу для всех шрифтов:

```
\setsansfont{Lato-Medium}[ Scale = MatchLowercase,
            Extension   = .ttf,
            Path        = /home/user/Fonts,
            UprightFont = Lato-Medium, ... ]
```

Если файлы используемых шрифтов находятся в папке, недоступной поисковой системе дистрибутива TEX, с помощью параметра Path нужно указать путь к ним.

Как видно из примеров, настройки задаются в виде списка параметров и их значений, разделяемых знаком равенства. Элементы списка разделяются запятыми, порядок их следования произволен. Параметры можно перечислять в общей строке, но для наглядности лучше разместить каждый на отдельной строке. Значения параметров, в свою очередь, могут быть списками, заключенными в фигурные скобки:

```
ItalicFont = {Font = Lato-MediumItalic, Style = Swash}.
```

Пакет fontspec позволяет составлять гарнитуры, объединяющие большое количество шрифтов. Описания их настроек могут быть довольно громоздкими, поэтому размещать их в параметрах команд, подключающих шрифты, неудобно. Для этого предназначены специальные декларации

```
\defaultfontfeatures[гарнитура]{настройки},
\defaultfontfeatures+[гарнитура]{настройки}.
```

Они различаются лишь тем, что первая формирует настройки заново, а вторая добавляет их к уже имеющимся. Данные декларации можно применять многократно в любом месте рукописи. Если параметр опущен, изменяются все гарнитуры, поэтому декларация `\defaultfontfeatures{}` с пустым параметром и аргументом позволяет сбросить все настройки, заданные различными пакетами, в том числе и самим fontspec.

В параметре деклараций можно указать названия гарнитур или файлов, а также команды, активирующие стандартные гарнитуры. Гарнитуры, имеющие общие настройки, можно перечислить списком, например конфигурационный файл содержит декларации

```
\defaultfontfeatures[\rmfamily, \sffamily]{Ligatures=TeX}
\defaultfontfeatures[\ttfamily]{%
        HyphenChar=None, WordSpace=1, PunctuationSpace=WordSpace}.
```

Первая инициирует использование лигатур TEX гарнитурами Serif и SansSerif, которые добавляются к лигатурам, имеющимся в самих шрифтах. Вторая запрещает переносы и устанавливает фиксированную ширину пробела для гарнитуры Mono (см. далее).

Еще одна декларация добавляет настройки к настройкам активной гарнитуры:

```
\addfontfeatures{настройки}.
```

Например, так можно установить для нее старый стиль печати чисел:

```
\addfontfeatures{Numbers=OldStyle}
```

Активация другой гарнитуры сбрасывает эти настройки.

С помощью условной конструкции

```
\IfFontFeatureActiveTF{параметр}{да}{нет}
```

можно проверить значение той или иной настройки активной гарнитуры, и в зависимости от результата выполнить те или иные действия:

```
\IfFontFeatureActiveTF{Numers=OldStyle}{}{%
                                \addfontfeatures{Numbers=OldStyle}}
```

Настройки гарнитур можно разместить в преамбуле рукописи, но так как шрифты используются в верстке многих документов, наиболее удобно загружать их из файлов. Подключая гарнитуру, fontspec ищет ее настройки в файле с именем, состоящим из ее названия, указанного в аргументе команд типа `\set...` или `\new...`, и расширения •.fontspec. Имена файлов с описаниями гарнитур могут отличаться от названий самих гарнитур. Например, декларацию `\defaultfontfeatures[XLato]{...}` с настройками шрифтов Lato можно поместить в файл с именем XLato.fontspec и подключать гарнитуру декларацией `\setsansfont{XLato}[...]`.

Приступим к анализу настроек. OpenType шрифты имеют их более сотни, и большинство из них fonspec поддерживает. Для однотипных настроек он вводит параметры, значениями которых служат OpenType настройки. Например, параметр Numbers устанавливает старый — OldStyle (ONUM), или обычный — Lining (LNUM), стиль цифр, а также моноширинный — Monospaced (TNUM), или пропорциональный — Proportional (PNUM), формат их вывода. Его значение SlashedZero (ZERO) выводит перечеркнутые нули, что удобно при распечатке кода программ. Здесь и далее в скобках капителью приведены обозначения настроек в самих шрифтах, их соответствие значениям параметров fontspec, выделенных рубленым шрифтом, описано в документации пакета [86].

В табл. 12.6 приведен список параметров пакета fonpec, содержащий более семидесяти наименований. Часть из них нам уже знакома, среди остальных выделим лишь те, что используются наиболее часто. Назначение остальных параметров описано в документации fontspec [86].

Таблица разделена на две половины. В верхней ее части перечислены параметры, предназначенные для подключения и трансформации шрифтов. Каждый из них можно настроить по-своему. Параметры левой колонки устанавливают шрифты, обеспечивающие стандартные начертания и контрастности. В простейшем случае их значениями являются имена файлов, но может быть и список, который должен содержать параметр Font с именем файла, например:

```
ItalicFont = { Font = Calibri_Italic, Style = ProportionalKana }.
```

Длинные списки удобней выносить в отдельные параметры, указанные во второй колонке:

```
ItalicFont     = Calibri\_Italic,
ItalicFeatures = { Style    = ProportionalKana,
                   CJKShape = Traditional,
                   Kerning  = On,  ... }.
```

Таблица 12.6

Параметры настройки OpenType гарнитур

UprightFont,	UprightFeatures,	Font,
ItalicFont,	ItalicFeatures,	FontFace,
BoldFont,	BoldFeatures,	FakeBold,
BoldItalicFont,	BoldItalicFeatures,	AutoFakeBold,
SlantedFont	SlantedFeatures,	FakeSlant,
BoldSlantedFont,	BoldSlantedFeatures,	AutoFakeSlant,
SwashFont,	SwashFeatures,	FakeStretch,
BoldSwashFont,	BoldSwashFeatures,	Size,
SmallCapsFont,	SmallCapsFeatures,	SizeFeatures,
Alternate,	Kerning,	PunctuationSpace,
Annotation,	Letters,	RawFeature,
CharacterVariant,	LetterSpace,	Renderer,
CharacterWidth,	Ligatures,	Scale,
Contextuals,	LocalForms,	ScaleAgain,
CJKShape,	NFSSEncoding,	Script,
Color,	NFSSFamily,	Style,
Diacritics,	Numbers,	StylisticAlternates,
Extension,	Opacity,	StylisticSet,
FontIndex,	OpticalSize,	Vertical,
Fractions,	Ornament,	VerticalPosition,
HyphenChar,	Path,	WordSpace.

Параметр FontFace позволяет подключить шрифт, отличный от тех, что перечислены в левой колонке. Например, среди них отсутствует разреженная капитель, определим ее следующим образом:

```
FontFace = {m}{ssc}{ Font    = Calibri_Regular,
                     Letters = SmallCaps, LetterSpace = 2}
```

Параметр Letters подключит капитель, содержащуюся в шрифте Calibri_Regular.ttf, а параметр LetterSpace вставит промежуток между буквами. Его значение дается в процентах размера шрифта, в нашем примере буквы кегля 10pt будет разделять промежуток шириной 0.2pt. Два первых «аргумента» параметра FontFace задают обозначение контрастности и начертания шрифта. Если они соответствуют стандартным аббревиатурам, шрифты будут автоматически подключать перечисленные в табл. 2.6 команды «уровня пользователя», в нашем случае это `\textssc{•}` и `\sscshape`. Шрифты с нестандартными обозначениями контрастности и начертания fontspec автоматически добавит в NFSS. Для их активации придется использовать команды низкого уровня. Например, вычурное начертание, обычно привязываемое к курсивному

шрифту, для прямого шрифта не является стандартным. Если добавить его в гарнитуру как `FontFace={m}{nw}{...}`, то шрифт можно будет подключать набором команд `\fontshape{nw}\selectfont`.

Шрифты нужного наклона, ширины или насыщенности, отсутствующие в гарнитуре, можно изготовить, трансформируя глифы имеющихся шрифтов. Трансформация снижает качество шрифтов, поэтому ее лучше не применять. Тем не менее параметры FakeSlant и FakeStretch позволяют изменить наклон и ширину глифов, а FakeBold — увеличить их насыщенность. Значением параметров является десятичное число, задающее коэффициент, на который изменяется ширина глифов или толщина образующих их линий. При значении FakeStretch, меньшем единицы, глифы сужаются. В случае наклона число означает тангенс угла наклона глифов. Отрицательное значение FakeSlant наклоняет глифы влево (например, для выпрямления курсива), а положительное — вправо. Используя параметр FakeSlant, сделаем наклоненный шрифт: `FontFace={m}{sl}{Font=•, FakeSlant=0.2}`. При использовании параметра AutoFakeSlant = • курсивный и наклонный шрифты автоматически генерируются из прямого шрифта, даже если они есть в самой гарнитуре, а параметр AutoFakeBold = • автоматически генерирует жирный шрифт.

Параметры Size и SizeFeatures позволяют настраивать шрифты разного размера. В качестве примера составим гарнитуру рубленых шрифтов New Computer Modern, у которых шрифты нормальной насыщенности разработаны в размерах 8pt и 10pt, а у жирных есть только кегль 10pt. Учтем это при составлении гарнитуры:

```
\defaultfontfeatures[NewCM]{
   Extension       = .otf ,
   UprightFont     = *10-Regular,
   UprightFeatures = { SizeFeatures = {
                       { Size = -9,  UprightFont = *08-Regular},
                       { Size = 9-,  UprightFont = *10-Regular} } },
   ItalicFont      = *10-Italic,
   ItalicFeatures  = { SizeFeatures = {
                       { Size = -9,  Font = *08-Italic},
                       { Size = 9-,  Font = *10-Italic} } },
   BoldFont        = *10-Bold,
   BoldItalicFont  = *10-BoldItalic }
```

Здесь мы применили сокращение имен файлов шрифтов, допустимое в описаниях гарнитур. Звездочка заменяет их часть, общую для

всех файлов, указанную в параметре команды `\defaultfontfeatures` в качестве названия гарнитуры. Таким образом, полное имя основного шрифта гарнитуры — NewCMSans10-Regular.otf, а жирного курсива — NewCMSans10-BoldOblique.otf.

Параметры SizeFeatures формируют шрифты в соответствии со значениями параметров Size, в которых дефис задает диапазоны размеров: –• означает *до* •pt, исключая верхнее значение; •–•• означает *от* •pt *до* ••pt, исключая верхнее значение; •– означает *от* •pt, включая нижнее значение. В нашем случае шрифты размером менее 9pt строятся из кегля 8pt, а начиная с этого размера — из кегля 10pt.

В верхней части табл. 12.6 имеются параметры Slanted... и SmallCaps..., позволяющие задать наклонные шрифты и капитель. Если наклонные шрифты не описаны, вместо них используется курсив. Если же не описана капитель, она будет доступна автоматически. Дело в том, что коды обычных строчных букв и капители латиницы различаются в кодировке unicode, поэтому шрифты могут содержать и те, и другие символы, а капитель других алфавитов может входить в них как стиль начертания строчных букв. По умолчанию используются обычные буквы, а параметр Letters = SmallCaps активирует использование капители (см. пример на с. 330). Пакет fontspec автоматически подключает капитель прямых, наклонных и курсивных шрифтов, если они ее содержат. Параметр SmallCapsFont определяет для нее альтернативный шрифт и отменяет использование других шрифтов, и даже если в них есть наклонная, курсивная и жирная капитель, команды `\textsc{•}` и `\scshape` не могут ее активировать.

Установка стиля вывода капители и заглавных букв сведена в параметр Letters, принимающий одно из следующих значений:

Uppercase (CASE) — заглавные буквы высотой со строчную d;

UppercasePetiteCaps (C2PC) — заглавные буквы высотой со строчную x;

UppercaseSmallCaps (C2SC) — заглавные буквы высотой как у строчных букв капители;

SmallCap (SMCP)s — обычная капитель;

PetiteCaps (PCAP) — капитель с высотой строчных букв x;

Unicase (UNIC) — гибрид строчных и прописных букв одинаковой высоты.

Параметр Letter, указанный в параметре SmallCapsFeatures, установит стиль капители всех шрифтов.

Шрифты OpenType поддерживают печать текста на многих языках. Они содержат не только глифы букв и иероглифов, но и информацию об их связи между собой, типа лигатур и кернинга, а также специфические настройки или скрипты, предназначенные для работы с определенными языками. Скрипт позволяет загрузить только ту часть ресурсов шрифта, которая действительно необходима для работы с тем или иным языком. Он определяет используемые алфавиты, операции верстки, совершаемые на уровне шрифта, наличие или отсутствие некоторых начертаний и даже некоторые настройки печати. Для печати русского текста нет нужды в иероглифах, и наоборот, для работы с японским текстом не нужна кириллица. Используемый скрипт должен соответствовать языкам, на которых написан текст. Список скриптов, поддерживаемых OpenType шрифтом, выводит на консоль программа otfinfo, запускаемая с ключом `-s`.

Пакет fontspec настраивает рабочий скрипт с помощью параметра Script. При верстке текста, набранного латиницей или кириллицей, его обычно можно опустить, но если какие-то начертания шрифтов отсутствуют в кириллице, можно попробовать установить ее скрипт: Script = Cyrillic . Тонкие настройки языка дает параметр Language, используемый в основном для восточных языков и иероглифического письма.

Уже упоминавшийся параметр Scale устанавливает для шрифта масштабирующий коэффициент, по умолчанию равный единице. Параметр ScaleAgain позволяет корректировать его, умножая текущее значение на коэффициент, заданный ScaleAgain. Такую коррекцию можно делать многократно.

Параметр Style задает стиль начертания букв. Среди его значений есть курсивы Italic (ITAL) и Cursive (CURS), вычурный шрифт Swash (SWSH) и несколько стилей вывода японских алфавитов кана. Используя параметры Letters и Style, можно сформировать вычурную капитель:

```
FontFace={m}{scsw}{Font=•, Letters=SmallCaps, Style=Swash}.
```

Параметр LetterSpace вставляет промежуток между буквами. Его значение задается в процентах размера шрифта. Используя параметры Letters и LetterSpace, можно сформировать разреженную капитель:

```
FontFace={m}{ssc}{Font=•, Letters=SmallCaps, LetterSpace=2}.
```

Параметры WordSpace и PunctuationSpace регулируют пробелы между словами и после знаков пунктуации, которые в шрифтах заданы упругими длинами. Они задают тройку чисел, на которые умножа-

ются значение длины, ее растяжимость и сжимаемость. Например, настройка WordSpace={1,0,0} или WordSpace=1 сделает длину пробела жесткой. Такое значение параметров WordSpace и PunctuationSpace обычно устанавливается в гарнитурах Mono.

Параметр HyphenChar устанавливает символ знака переноса, но главное его назначение — подавление переносов, задаваемое специальным значением HyphenChar=None. Как правило, переносы подавляются в гарнитурах Mono.

Параметры Color и Opacity делают шрифты цветными и полупрозрачными. Для их использования нужно загрузить пакет xcolor. Цвета можно задать названиями, например Color = blue. У основных цветов они определены по умолчанию, а названия остальных зависят от параметров загрузки пакета xcolor [55]. Прозрачность задается десятичным числом от нуля (полностью прозрачный) до единицы (непрозрачный).

Параметр RawFeature позволяет активировать любые настройки OpenType гарнитур. Его значениями являются непосредственно OpenType настройки, полный список которых содержит документация пакета fontspec [86]. Там же обстоятельно описаны и все те параметры, обсуждение которых мы опустили.

Два последних параметра оптимизируют использование OpenType шрифтов в LaTeX. Ранее говорилось, что fontspec вводит для подключения дополнительных гарнитур *нестандартные* декларации. В то же время в настройках гарнитур предусмотрен параметр NFSSFamily, присваивающий им идентификаторы системы NFSS. Он позволяет избежать использование нестандартных деклараций. Например, гарнитуру настройкой NFSSFamily = MyFont можно подключить декларацией NFSS `\fontfamily{MyFont}\selectfont`.

Параметр NFSSEncoding предназначен для обработки символов, заданных в рукописях авторов командами. Как уже отмечалось, пакет fontspec загружает с этой целью кодировку TU. Команда

`\DeclareUnicodeEncoding{`*название*`}{`*список кодировок*`}`

в сочетании с параметром NFSSEncoding = *название* дает возможность расширить список загружаемых кодировок. *Название*, используемое в команде и параметре, должно быть уникальным, а *список* должен содержать файл tuenc.def с кодировкой TU. За дальнейшими разъяснениями мы вновь отсылаем читателя к документации пакета fontspec [86].

12.3.5. Примеры настройки шрифтов

Автор может настроить гарнитуры по-своему и пользоваться ими, но так как настройки многих из них, уже имеются в дистрибутивах TEX, чтобы избежать коллизии с их описаниями, к названиям гарнитур, обсуждаемым далее, мы добавили префикс X. Описания гарнитур удобно разместить в файлах с расширением •.fontspec, которые должны находиться не в папках со шрифтами, а в папках с пакетами (см. рис. А.2). Имя файла должно совпадать с названием гарнитуры, тогда автоматически будут доступны прямой и курсивный шрифты вместе с капителью. Вычурные шрифты нужно определять особо, вместе с ними станет доступна и вычурная капитель. Разреженную капитель можно сделать, вставив небольшой пробел между буквами с помощью параметра LetterSpace. Это не ухудшит качество шрифтов, так как не исказит их глифы. Стиль цифр в создаваемых гарнитурах лучше не указывать, тогда они будут печататься в обычном стиле. В старом стиле их всегда можно напечатать стандартной командой `\oldstylenums`.

Сделав эти предварительные замечания, приступим к расширению возможностей обсуждавшихся ранее гарнитур и, прежде всего, отметим, что гарнитуры Anonymous Pro и Courier New этого не требуют, так как все их шрифты доступны при подключении по названию гарнитуры.

Гарнитура XLato

Обсудим для начала, как составлена показанная на рис. 12.6 гарнитура XLato, объединяющая все шрифты Lato:

```
\defaultfontfeatures[XLato]{
   Ligatures=       TeX,
   Extension=      .ttf,
   NFSSFamily=      XLato,
   UprightFont=     Lato-Medium,
   ItalicFont=      Lato-MediumItalic,
   BoldFont=        Lato-Bold,
   BoldItalicFont=  Lato-BoldItalic,
   FontFace={ul}{n} {Font=Lato-Hairline},
   FontFace={ul}{it}{Font=Lato-HairlineItalic},
   FontFace={el}{n} {Font=Lato-Thin},
   FontFace={el}{it}{Font=Lato-ThinItalic},
   FontFace={l}{n}  {Font=Lato-Light},
   FontFace={l}{it} {Font=Lato-LightItalic},
   FontFace={sl}{n} {Font=Lato-Regular},
```

```
   FontFace={sl}{it}{Font=Lato-Italic},
   FontFace={sb}{n} {Font=Lato-Semibold},
   FontFace={sb}{it}{Font=Lato-SemiboldItalic},
   FontFace={eb}{n} {Font=Lato-Heavy},
   FontFace={eb}{it}{Font=Lato-HeavyItalic},
   FontFace={eb}{n} {Font=Lato-Black},
   FontFace={eb}{it}{Font=Lato-BlackItalic} }
```

Параметр Ligatures требуется для обработки лигатур при подключении XLato в качестве как дополнительной гарнитуры, параметр NFSSFamily позволит активировать ее стандартными командами NFSS `\fontfamily{XLato}\selectfont`, а параметр Extension обеспечит однозначный выбор файлов шрифтов.

В гарнитуре Lato есть только прямые и курсивные начертания, капитель и вычурные шрифты отсутствуют. Полусветлая (`sl`) и средняя (`m`) контрастности представлены шрифтами насыщенности Regular и Medium. Такой выбор удобен при комбинировании XLato с контрастными гарнитурами типа Garamond, STIX2 и т. п.[8] Напомним, что на с. 325 обсуждались настройки команды `\strong`, дающие доступ ко всем жирными шрифтам гарнитуры XLato.

Гарнитура XEBGaramond

При подключении гарнитуры EBGaramond способом, описанным на с. 303, значительное количество ее шрифтов оказывается недоступно. Полный набор шрифтов представлен на рис. 12.7. В его нижней части для сравнения приведены шрифты разной контрастности, каждый из которых обладает всеми начертаниями, показанными в его верхней части. Вот как составлена гарнитура XEBGaramond, содержащая эти шрифты:

```
\defaultfontfeatures[XEBGaramond]{
   Ligatures=      TeX,
   Extension=     .otf,
   NFSSFamily=     XEBGaramond,
   UprightFont=    EBGaramond-Regular,
   ItalicFont=     EBGaramond-Italic,
   BoldFont=       EBGaramond-SemiBold,
   BoldItalicFont= EBGaramond-SemiBoldItalic,
```

[8] Для работы со светлыми гарнитурами типа Computer Modern лучше подойдут настройки пакета lato, устанавливающего в качестве основных более светлые шрифты Regular. Для шрифтов насыщенности Medium он вводит нестандартную контрастность `mb`.

Гарнитура XEBGaramond

m:n	Vita sine libertāti, nihil.	Жизнь без свободы — ничто.
m:it	*Vita sine libertāti, nihil.*	*Жизнь без свободы — ничто.*
m:sc	Dura lex, sed lex.	Суров закон, но это закон.
m:ssc	Dura lex, sed lex.	Суров закон, но это закон.
m:scit	*Dura lex, sed lex.*	*Суров закон, но это закон.*
m:sw	*A B C D E F G H I J K L M N O P Q R T S U V W X Y Z*	
	Necessitas frangit legem.	*Необходимость ломает закон.*
m:scsw	*Necessitas frangit legem.*	*Необходимость ломает закон.*
b:n	**Potius sero quam nunguam.**	**Лучше поздно, чем никогда.**
b:it	***Potius sero quam nunguam.***	***Лучше поздно, чем никогда.***
b:sc	**Omnia vincit amor.**	**Любовь побеждает все.**
b:ssc	**Omnia vincit amor.**	**Любовь побеждает все.**
b:scit	***Omnia vincit amor.***	***Любовь побеждает все.***
b:sw	***A B C D E F G H I J K L M N O P Q R T S U V W X Y Z***	
	Credo, quia absurdum.	***Верю, ибо это абсурдно.***
b:scsw	***Credo, quia absurdum.***	***Верю, ибо это абсурдно.***

m:n	Vana est sapientia nostra.	Тщетна наша мудрость.
sb:n	Vana est sapientia nostra.	Тщетна наша мудрость.
b:n	**Vana est sapientia nostra.**	**Тщетна наша мудрость.**
eb:n	**Vana est sapientia nostra.**	**Тщетна наша мудрость.**
ub:n	**Vana est sapientia nostra.**	**Тщетна наша мудрость.**

Рис. 12.7. Шрифты гарнитуры XEBGaramond

```
FontFace={m}{sw}  {Font=EBGaramond-Italic, Style=Swash},
FontFace={m}{ssc} {Font=EBGaramond-Regular,
                   Letters=SmallCaps, LetterSpace=1},
FontFace={sb}{n}  {Font=EBGaramond-Medium},
FontFace={sb}{it} {Font=EBGaramond-MediumItalic},
.....
FontFace={eb}{n}  {Font=EBGaramond-Bold},
FontFace={eb}{it} {Font=EBGaramond-BoldItalic},
.....
FontFace={ub}{n}  {Font=EBGaramond-ExtraBold},
FontFace={ub}{it} {Font=EBGaramond-ExtraBoldItalic}, ..... }
```

Средняя (m) и жирная (b) контрастности гарнитуры XEBGaramond представлены шрифтами Regular и Semibold, в качестве полужирных (sb) и очень жирных (e, ub) используются шрифты контрастности Medium, Bold и ExtraBold. На примере шрифтов средней насыщенности показано определение вычурного шрифта (sw) и разреженной капители (ssc). Для остальных контрастностей они вводятся совершенно так же, поэтому их определения опущены. Остальные начертания подключаются автоматически.

Для работы с жирными шрифтами можно пользоваться командой \strong, настроенной аналогично предыдущему случаю.

Гарнитура XSTIX2

При подключении гарнитуры STIX2 пакетом fontsetup доступны шрифты только двух из четырех контрастностей, показанных на рис. 12.8. Приведем описание гарнитуры XSTIX2, содержащей все эти шрифты:

```
\defaultfontfeatures[XSTIX2]{
   Script=         Default,
   NFSSFamily=     XSTIX2,
   Extension=     .otf,
   UprightFont=    STIXTwoText-Regular,
   ItalicFont=     STIXTwoText-Italic,
   FontFace={m}{ssc} {Font=STIXTwoText-Regular, Letters=SmallCaps,
                      LetterSpace=2},
   BoldFont=       STIXTwoText-SemiBold,
   BoldItalicFont= STIXTwoText-SemiBoldItalic,
   .....
   FontFace={sb}{n} {Font= STIXTwoText-Medium},
   FontFace={sb}{it}{Font= STIXTwoText-MediumItalic},
   .....
   FontFace={eb}{n} {Font= STIXTwoText-Bold},
   FontFace={eb}{it}{Font= STIXTwoText-BoldItalic}, ..... }
```

Как и в предыдущем случае, шрифты разной контрастности, собранные в его нижней части, обладают всеми начертаниями, показанными в верхней части. Набор начертаний включает прямой и курсивный шрифты с различными вариантами капители. Вычурных шрифтов нет.

Обратим внимание на наличие параметра Script = Default в описании гарнитуры XSTIX2. Он требуется, чтобы вывести капитель кириллицы. Без него капитель поддерживает только латиницу, а русские буквы остаются печатными.

Гарнитура XSTIX2

`m:n`	Nil permanet sub sole.	Ничто не вечно под солнцем.
`m:it`	*Nil permanet sub sole.*	*Ничто не вечно под солнцем.*
`m:sc`	FÓRTES FÓRTUNA ÁDJUVAT.	Смелым помогает судьба.
`m:ssc`	FÓRTES FÓRTUNA ÁDJUVAT.	Смелым помогает судьба.
`m:scit`	*FÓRTES FÓRTUNA ÁDJUVAT.*	*Смелым помогает судьба.*
`b:n`	**Magna res est amor.**	**Великая вещь — любовь.**
`b:it`	***Magna res est amor.***	***Великая вещь — любовь.***
`b:sc`	**SPIRITUS FLAT UBI VULT.**	**Дух веет, где хочет.**
`b:ssc`	**SPIRITUS FLAT UBI VULT.**	**Дух веет, где хочет.**
`b:scit`	***SPIRITUS FLAT UBI VULT.***	***Дух веет, где хочет.***
`m:n`	Natūra abhorret vacuum.	Природа боится пустоты.
`sb:n`	**Natūra abhorret vacuum.**	**Природа боится пустоты.**
`b:n`	**Natūra abhorret vacuum.**	**Природа боится пустоты.**
`eb:n`	**Natūra abhorret vacuum.**	**Природа боится пустоты.**

Рис. 12.8. Шрифты гарнитуры XSTIX2

Аналогично описанию гарнитуры XEBGaramond, определение разреженной капители показано только для шрифта средней насыщенности. В других шрифтах она определена так же, а остальные начертания подключаются автоматически.

Как и в двух предыдущих случаях, для работы с жирными шрифтами можно воспользоваться настроенной командой `\strong`.

Гарнитура Times New Roman

Шрифты Times New Roman имеют только две контрастности, но при этом в нынешней версии 7.00 располагают одним из самых полных наборов прекрасно прорисованных начертаний, показанных на рис. 12.9. Все они доступны в гарнитуре XTimesNewRoman:

```
\defaultfontfeatures[XTimesNewRoman]{
  Ligatures=      TeX,
  UprightFont=    times,
  ItalicFont=     timesi,
  BoldFont=       timesbd,
  BoldItalicFont= timesbi,
  FontFace={m}{sw} {Font=timesi, Style=Swash},
```

Гарнитура XTimesNewRoman

m:n	Puris omnia pura.	Для чистых все чисто.
m:it	*Puris omnia pura.*	*Для чистых все чисто.*
m:sc	Quaerite et invetiētis.	Ищите и обрящете.
m:ssc	Quaerite et invetiētis.	Ищите и обрящете.
m:scit	*Quaerite et invetiētis.*	*Ищите и обрящете.*
m:sw	*A B C D E F G H I J K L M N O P Q R T S U V W X Y Z*	
	Sic transit gloria mundi.	*Так проходит мирская слава.*
m:scsw	*Sic transit gloria mundi.*	*Так проходит мирская слава.*
b:n	**Scio me nihil scire.**	**Я знаю, что ничего не знаю.**
b:it	***Scio me nihil scire.***	***Я знаю, что ничего не знаю.***
b:sc	**Nosce te ipsum.**	**Познай самого себя.**
b:ssc	**Nosce te ipsum.**	**Познай самого себя.**
b:scit	***Nosce te ipsum.***	***Познай самого себя.***
b:sw	*A B C D E F G H I J K L M N O P Q R T S U V W X Y Z*	
	Memento mori.	***Помни о смерти.***
b:scsw	***Memento mori.***	***Помни о смерти.***

Рис. 12.9. Шрифты гарнитуры XTimesNewRoman

```
FontFace={m}{ssc}{Font=times, Letters=SmallCaps, LetterSpace=3},
FontFace={b}{sw} {Font=timesbi,Style=Swash},
FontFace={b}{ssc}{Font=timesbd,Letters=SmallCaps, LetterSpace=3} }
```

В описаниях гарнитур имена файлов со шрифтами нужно задавать *однозначно*, однако в разное время шрифты Times New Roman назывались по-разному. Наше описание использует «универсальные» имена, введенные в ранних версиях ОС Windows. В целях унификации они продолжают поддерживаться как современными версиями ОС Windows, так и другими операционнымие системами.

Все шрифты гарнитуры XTimesNewRoman доступны с помощью стандартных команд, приведенных в табл. 2.6.

Гарнитура XCalibri

Помимо шрифтов стандартных контрастностей гарнитура Calibri содержит показанные на рис. 12.9 светлые шрифты, недоступные при ее подключении, описанном на с. 308. Мы добавили их в гарнитуру XCalibri (рис. 12.10):

Гарнитура XCalibri

l:n	Acta est fabula.	Представление окончено.
l:sc	Acta est fabula.	Представление окончено.
l:ssc	Acta est fabula.	Представление окончено.
m:n	Conscientia mille testes.	Совесть — тысяча свидетелей.
m:it	*Conscientia mille testes.*	*Совесть — тысяча свидетелей.*
m:sc	Factum est factum.	Что сделано, то сделано.
m:ssc	Factum est factum.	Что сделано, то сделано.
m:scit	*Factum est factum.*	*Что сделано, то сделано.*
b:n	**Dum spiro, spero.**	**Пока дышу, надеюсь.**
b:it	***Dum spiro, spero.***	***Пока дышу, надеюсь.***
b:sc	**Nulla salus bello.**	**Нет блага в войне.**
b:ssc	**Nulla salus bello.**	**Нет блага в войне.**
b:scit	***Nulla salus bello.***	***Нет блага в войне.***

Рис. 12.10. Шрифты гарнитуры XCalibri

```
\defaultfontfeatures[XCalibri]{
   Ligatures=      TeX,
   UprightFont=    Calibri_Regular,
   ItalicFont=     Calibri_Italic,
   FontFace={m}{ssc}{Font=Calibri_Regular, Letters=SmallCaps,
                     LetterSpace=2}, .....
   BoldFont=       Calibri_Bold,
   BoldItalicFont= Calibri_Bold_Italic, .....
   FontFace={l}{n}  {Font=Calibri_Light},
   FontFace={m}{ssc}{Font=Calibri_Regular, Letters=SmallCaps,
                    LetterSpace=2}, ..... }
```

Аналогично предыдущим случаям, определение разреженной капители показано только для шрифта средней насыщенности. В других шрифтах она определена так же, а обычная капитель подключается автоматически.

Для активации светлых шрифтов имеет смысл настроить декларацию `\strongreset`:

```
\renewcommand\strongreset{\fontseries{l}\selectfont}.
```

Вместе с другими командами, изменяющими контрастность шрифтов, она позволит реализовать все возможности гарнитуры Calibri.

12.3.6. Пакет unicode-math

OpenType технологии дают богатые возможности в верстке формул. Основные их преимущества заключаются в простоте настройки математических шрифтов и возможности ввода математических символов без использования команд, сильно упрощающей набор формул и делающей их код намного нагляднее. Символы формул, сверстанных с их помощью, адекватно воспроизводятся при копировании, в частности в рукопись можно копировать формулы из документа Microsoft Word и использовать их после небольшой доработки. Программа TEXstudio легко настраивается на ввод символов из графического меню (см. с. 13).

Приведем примеры формулы, сверстанной OpenType шрифтами:

```
\setmathfont{NewComputerModernMath-Regular}[math-style=literal]
\setmathfont{XITSMath-Bold}[version=bold]
```

`$\frac12 VH^2_c = -4\pi\int_0^{H_c} \mathbf{M}\,d\mathbf{H}$,` $\xrightarrow{\text{literal}}$ $\frac{1}{2} V H_c^2 = -4\pi \int_0^{H_c} \mathbf{M}\, d\mathbf{H}$,

`\boldmath $...$,` $\xrightarrow{\text{bold, literal}}$ $\boldsymbol{\frac{1}{2} V H_c^2 = -4\pi \int_0^{H_c} \mathbf{M}\, d\mathbf{H}}$,

```
\setmathfont{NewComputerModernMath-Regular}[math-style=upright]
```

`$\frac12 VH^2_c = -4\pi\int_0^{H_c} \mathbf{M}\,d\mathbf{H}$,` $\xrightarrow{\text{upright}}$ $\frac{1}{2} \mathrm{V H_c^2} = -4\pi \int_0^{\mathrm{H_c}} \mathbf{M}\, \mathrm{d}\mathbf{H}$,

```
\setmathfont{Garamond Math}[version=Garamond]
```

`\mathversion{Garamond} $...$,` $\xrightarrow{\text{upright}}$ $\frac{1}{2} \mathrm{V H_c^2} = -4\pi \int_0^{\mathrm{H_c}} \mathbf{M}\, \mathrm{d}\mathbf{H}$,

```
$\frac12 H^2_c = -4\pi\int_0^{H_c} \mathbf{M}\,d\mathbf{H}$.
```

Вверху слева показан код формулы LaTeX, а справа — различные варианты ее верстки. Видно, что в двух первых примерах компиляция воспроизводит символы «в естественном виде»: курсивные буквы становятся математическим курсивом, прямые остаются прямыми. В двух последних примерах после компиляции все буквы становятся прямыми. Внизу приведено самое компактное представление кода для верстки компилятором PDFLaTeX, далеко проигрывающее в наглядности коду, верстаемому компиляторами LuaLaTeX и XeLaTeX, которые с равным успехом справятся с обоими вариантами кода. OpenType технологии обеспечивают простой и удобный доступ к жирным шрифтам и другим математическим гарнитурам. Наш пример демонстрирует использование дополнительной гарнитуры Garamond Math и жирного шрифта XITSMath-Bold.

Уже сейчас имеется значительное число математических OpenType гарнитур, и большинство из них поддерживает unicode-math — пакет, встраивающий их в LaTeX [87]. В настоящее время он работает со свободно распространяемыми шрифтами Asana-Math (asana-math), Fira Math (firamath-otf) [93], Garamond-Math (garamond-math) [91], Latin Modern Math (lm-math), Libertinus Math (libertinus-fonts) [94], New Computer Modern Math (newcomputermodern), STIX Two Math (stix2-otf) [95], XITS Math (xits), шрифтами пакета tex-gyre-math TeX Gyre Bonum Math, TeX Gyre DejaVu Math, TeX Gyre Pagella Math, TeX Gyre Schola Math, TeX Gyre Termes Math, а также коммерческими шрифтами Lucida Bright Math, Cambria Math, Minion Math. В скобках указаны названия пакетов, устанавливающих соответствующие гарнитуры. Каждая из них насчитывает свыше тысячи символов, а XITS, STIX2 и New Computer Modern Math — почти две с половиной тысячи. Образцы шрифтов вместе с кодами и наименованиями символов в кодировке unicode приведены в документации пакета unicode-math [88]. И хотя он пока еще имеет экспериментальный статус, он полностью функционален. Его изменения в ходе доработки в основном коснутся деталей, которые вряд ли интересуют большинство авторов.

Загрузка пакета

Пакет unicode-math прост в использовании: достаточно загрузить его и задать основные гарнитуры *текста*. Его настройки по умолчанию соответствуют стилю верстки формул стандартного LaTeX, поэтому в большинстве случаев вносить изменения в набранные формулы не потребуется. Символы, присутствующие в них в виде команд, будут распознаны во время компиляции. Для этого unicode-math загружает обширную таблицу, связывающую команды LaTeX, введенные большим числом математических пакетов, с их кодами в OpenType шрифтах.

Для работы со шрифтами unicode-math автоматически загружает пакет fontspec, а для верстки математических конструкций пакет amsmath. К ним рекомендуется добавить пакет amssymb. Хотя любая математическая гарнитура OpenType поставляет символы с избытком, часть символов LaTeX, выделенная фоном в табл. 4.4, отсутствующая в кодировке unicode, окажется недоступна без него. Математические пакеты нужно загружать, соблюдая правильный порядок: amssymb и mathtools (с настройками amsmath) *до* unicode-math, а mathcd — *после*.

После загрузки unicode-math нужно задать математическую гарнитуру, хотя и это требование не обязательно. Если она не указана, формулы верстаются шрифтом Latin Modern Math.

Настройка верстки

В кодировке unicode некоторые символы имеют несколько стилей вывода. Самый яркий пример — латинские буквы, которые могут быть прямыми, курсивными, жирными, каллиграфическими, рубленными, моноширинными, готическими и и т. д. Чтобы компиляторы обработали такие символы правильно, нужно уточнить, какой набор глифов использовать при печати. Для этого предназначены настройки пакета unicode-math, приведенные в табл. 12.7 и 12.8.

В рукописи можно использовать как прямое, так и курсивное начертание букв и операторов ∂, ∇, в документе они будут напечатаны в соответствии с выбранным стилем формул. Пакет unicode-math реализует два алгоритма верстки: *символьный* и *стилевой*. Символьная верстка переводит коды символов в соответствующие им глифы шрифтов, как показывают первые два варианта верстки нашего примера. При стилевой верстке начертание латинских и греческих букв в рукописи не влияет на их представление в документе (см. два последних варианта). В зависимости от настроек они могут печататься как прямым шрифтом, так и курсивом. Символы математических операций, соотношений и т. п. однозначно связаны с глифами OpenType шрифтов, поэтому их печать не зависит от алгоритма верстки.

Список настроек пакета unicode-math приведен в табл. 12.7. В левой колонке перечислены параметры, а в правой значения, которые они могут принимать. В верхней части таблицы выше отчеркивающей линии представлены настройки самого пакета, а в ее нижней части — параметры, которые он добавляет к настройкам OpenType гарнитур. Жирным шрифтом выделены их значения по умолчанию.

В простейшем случае формулы верстаются одним OpenType шрифтом, однако пакет unicode-math позволяет собрать математическую гарнитуру из разных шрифтов и, более того, использовать несколько таких гарнитур. Для создания гарнитур предназначены настройки из нижней части таблицы. «Сборные» гарнитуры используются в верстке, настраиваемой параметрами из верхней части таблицы, которые действуют на все гарнитуры, определяя, как и какие шрифты будут задействованы.

Таблица 12.7

Настройки пакета unicode-math

Параметры	Значения
bold-style	french, ISO, literal, **TeX**, upright
math-style	french, ISO, literal, **TeX**, upright
sans-style	italic, literal, **upright**
nabla	italic, literal, **upright**
partial	**italic**, literal, upright
colon	literal, **TeX**
slash-delimiter	**ascii**, div, frac
active-frac	normalsize, **small**
mathbf	sym, **text**
mathit	sym, **text**
mathrm	sym, **text**
mathsf	sym, **text**
mathtt	sym, **text**
version	bold, **normal**, *гарнитура*
script-font	*шрифт*
script-features	MathScript, ...
sscript-font	*шрифт*
sscript-features	MathScriptScript, ...
range	up, it, sf, sfit, frak, cal, scr, tt, bb, bbit, bfup, bfit, bfsf, bfsfit, bffrak, bfcal, bfscr, Greek, greek, Latin, latin, num, `"XXXX-"YYYY, \mathopen, ..., \int, ...`

Таблица 12.8

Настройки стиля формул

Параметры	ISO	**TeX**	french	upright
math-style	*abc ABC ∂* αβδ *ΓΔΩ* ∇ *abc ABC*	*abc ABC ∂* αβδ ΓΔΩ ∇ abc ABC	*abc* ABC ∂ αβδ ΓΔΩ ∇ abc ABC	abc ABC ∂ αβδ ΓΔΩ ∇ abc ABC
bold-style	***abc ABC ∂*** **αβδ *ΓΔΩ* ∇** ***abc ABC ∂*** ***αβδ ΓΔΩ ∇***	**abc ABC ∂** **αβδ ΓΔΩ ∇** **abc ABC ∂** **αβδ ΓΔΩ ∇**	***abc* ABC ∂** **αβδ ΓΔΩ ∇** **abc ABC ∂** **αβδ ΓΔΩ ∇**	**abc ABC ∂** **αβδ ΓΔΩ ∇** **abc ABC ∂** **αβδ ΓΔΩ ∇**

Рассмотрим сначала настройки пакета. Их можно задать при его загрузке, в команде `\unimathsetup{`*список параметров*`}` и даже в настройках гарнитур. Все три способа можно использовать вместе:

```
\usepackage[math-style=french]{unicode-math}
\unimathsetup{mathbf=sym, mathsf=sym, active-frac=normalsize}
\setmathfont{STIX Two Math}[partial=italic, ...]
```

Подчеркнем еще раз, что настройки пакета, указанные в настройках гарнитуры, применяются не только к ней, но и ко всем остальным. Это хорошо видно из приведенного выше примера. Сначала активация гарнитуры NewComputerModernMath-Regular устанавливает стиль верстки literal (см. далее), активация жирного шрифта XITSMath-Bold его не меняет, поэтому две первые формулы сверстаны в этом стиле. Последующая активация гарнитуры NewComputerModernMath-Regular задает стиль upright, «наследуемый» гарнитурой Garamond Math, и следующие формулы сверстаны в новом стиле.

Действие команды `\unimathsetup` зависит от типа настроек. Те из них, что выделены в табл. 12.7 фоном, начинают действовать сразу, их можно менять в любом месте рукописи. Для параметров, указанных вверху без фона, ситуация аналогична замене атрибута шрифта в NFSS. Команда `\unimathsetup` только объявляет их новые значения, которые нужно активировать декларацией `\setmathfont`. Но так как она сбросит сделанные ранее настройки гарнитуры, данные параметры нужно установить после загрузки пакета *до* активации гарнитуры, или сделать это в ее настройках.

Разобравшись с методами задания настроек, перейдем к анализу их действия. Параметры math-style и bold-style определяют начертания букв, или стиль формул. Если они не установлены явно, используется стиль TeX, а если явно задан только параметр math-style, его значение присваивается и bold-style. Стиль literal соответствует символьному алгоритму верстки, в котором начертание букв устанавливает сам автор, печатая их в нужном виде в рукописи. Образцы стилей, применяемых в стилевой верстке, показаны в табл. 12.8. Во всех стилях оператор градиента имеет прямое начертание. В стиле ISO остальные символы печатаются курсивом. В стиле upright все символы — прямые. В стиле french латинские строчные буквы, кроме рубленых, имеют курсивное начертание, а остальные символы — прямые. В стиле TeX прямым шрифтом печатаются все рубленые буквы, заглавные греческие буквы, а также заглавные жирные буквы с засечками. Остальные символы имеют курсивное начертание.

Параметры sans-style, nabla и partial позволяют изменить стиль печати рубленого шрифта, оператора градиента и дифференциала частных производных *после* установки стиля верстки.[9] Например, настройка nabla=italic сделает оператора градиента курсивным.

Команды `\symliteral{`*символы*`}` и `\symnormal{`*символы*`}` позволяют, соответственно, защитить от преобразования отдельные символы или, напротив, привести их в соответствие со стилем формулы.

Как бы ни были напечатаны буквы в рукописи, стилевая верстка приведет их к заданному виду. Уточним, что преобразованию подвергаются только буквы, содержащиеся в рукописи в виде символов, а те, что вводят команды математических алфавитов, остаются неизменными, за исключением алфавитов `\mathsf` и `\mathbf`, печать которых зависит от значений параметров mathsf и mathbf.

Настройка алфавитов

Шрифты OpenType содержат в себе множество математических алфавитов, представленных в табл. 12.9. В ее правой части показаны образцы их букв и цифр, а если ячейки пусты — соответствующие символы отсутствуют. Алфавиты могут содержать латинские и греческие буквы, цифры, дифференциал частных производных и оператор градиента. Два последних символа греческими буквами не являются, но в кодировке unicode находятся вместе с ними, поэтому мы также поместили их вместе.

Алфавиты, входящие в состав стандартных математических шрифтов LaTeX (см. разд. 4.1.1), печатают стандартные команды LaTeX из первой колонки. Пакет unicode-math дополняет их собственными командами, перечисленными во второй колонке. Большинство из них представлено парой синонимов с префиксами `\sym...` и `\math...`. Документация unicode-math [87] рекомендует пользоваться командами с именами, начинающимися с `\sym...`, так как в будущем некоторые синонимы могут быть изъяты. Следуя рекомендации, мы будем оперировать только ими. Отметим, что вложение команд, оперирующих алфавитами, не поддерживается, т. е. `\symbfsf{•}` ≠ `\symbf{\symsf{•}}`.

Среди стандартных шрифтов отсутствуют рубленый и ажурный курсивы, для которых пакет unicode-math вводит новые команды `\symsfit{•}`, `\symbfsfit{•}` и `\symbbit{•}`. Последний алфавит состоит их весьма ограниченного набора символов.

[9] Установка стиля верстки заменит ранее введенные настройки.

Команды \symbb, \symcal, \symfrak и \symscr служат синонимами стандартных команд \mathbb, \mathcal, \mathfrak и \mathscr, но остальные команды \sym... имеют самостоятельное значение.

Алфавиты, перечисленные в верхней части табл. 12.9, являются аналогами основных гарнитур текста Serif, SansSerif и Mono. Чтобы обеспечить единый стиль текста и формул, пакет fontspec заменяет алфавиты этими гарнитурами, и их изменение автоматически меняет и алфавиты. Если такая замена нежелательна, загрузите пакет с параметром no-math, тогда формулы будут верстаться только математическими шрифтами. Пакет fontspec предлагает также и альтернативу, вводя декларации, позволяющие настроить алфавиты по своему усмотрению:

```
\setmathrm{гарнитура}[настройки],
\setmathsf{гарнитура}[настройки],
\setmathtt{гарнитура}[настройки],
\setboldmathrm{гарнитура}[настройки]
```

Алфавиты, определенные таким образом, не зависят от основных гарнитур, их можно настраивать и менять многократно в любом месте рукописи. Отметим, что декларация \setmathrm устанавливает сразу три алфавита: *гарнитуру* в качестве \mathrm и ее курсив в качестве \mathit и жирный шрифт в качестве \mathbf, а декларация \setboldmathrm позволяет задать жирный алфавит отдельно.

Пакет unicode-math корректирует действия fontspec. Он связывает основные гарнитуры или шрифты, устанавливаемые декларациями \setmath..., с командами[10]

```
\mathtextrm{текст},  \mathtextsf{текст},
\mathtextit{текст},  \mathtexttt{текст},
\mathtextbf{текст}.
```

Они, в свою очередь, привязываются к алфавитам настройками math..., выделенными фоном в табл. 12.7. Значение sym делает команды алфавитов \math... синонимами команд \sym... В этом случае для печати используется математическая гарнитура. При используемом по умолчанию значении text алфавиты привязываются к основным гарнитурам, или шрифтам, установленным декларациями \setmath...

Ранее говорилось, что обработка алфавитов \mathbf и \mathsf при стилевой верстке зависит от настроек mathbf и mathsf. Их буквы не

[10] Это делается и при загрузке fontspec с параметром no-math.

Таблица 12.9

Математические алфавиты пакета unicode-math

LaTeX	Команды пакета unicode-math	Буквы				Цифры
		латинские		греческие		
\mathrm{•}	\symup{•}, \mathup{•}	a … z,	A … Z	α … ω, ∂	A … Ω, ∇	0 … 9
\mathbf{•}	\symbf{•}, \symbfup{•}	**a … z,**	**A … Z**	**α … ω, ∂**	**A … Ω, ∇**	**0 … 9**
\mathit{•}	\symit{•}	*a … z,*	*A … Z*	*α … ω, ∂*	*A … Ω, ∇*	
\bm{\mathit{•}}	\symbfit{•}, \mathbfit{•}	***a … z,***	***A … Z***	***α … ω, ∂***	***A … Ω, ∇***	
\mathsf{•}	\symsf{•}, \symsfup{•}, \mathsfup{•}	a … z,	A … Z			0 … 9
\bm{\mathsf{•}}	\symbfsfup{•}, \mathbfsfup{•}	**a … z,**	**A … Z**	**α … ω, ∂**	**A … Ω, ∇**	**0 … 9**
	\symsfit{•}, \mathsfit{•}	*a … z,*	*A … Z*			
	\symbfsfit{•}, \mathbfsfit{•}	***a … z,***	***A … Z***	***α … ω, ∂***	***A … Ω, ∇***	
\mathtt{•}	\symtt{•}	a … z,	A … Z			0 … 9
\mathbb{•}	\symbb{•}	a … z,	A … Z			0 … 9
	\symbbit{•}, \mathbbit{•}	*d*, *e*, *i*, *j*, *D*				
\mathfrak{•}	\symfrak{•}	a … z,	A … Z			
\bm{\mathfrak{•}}	\symbffrak{•}, \mathbffrak{•}	**a … z,**	**A … Z**			
\mathscr{•}	\symscr{•}	*a … z,*	*A … Z*			
\bm{\mathfscr{•}}	\symbfscr{•}, \mathbfscr{•}	***a … z,***	***A … Z***			
\mathcal{•}	\symcal{•}		*A … Z*			
\bm{\mathfcal{•}}	\symbfcal{•}, \mathbfcal{•}		***A … Z***			

меняются, если данные параметры установлены в значение text, «отключающее» алфавиты от математических шрифтов. В противном случае они преобразуются в соответствии со стилем формулы.

Настройки пакета unicode-math, отключающие алфавиты от математической гарнитуры, далеко не оптимальны, так как ее буквы встраиваются в формулу немного иначе чем буквы текста. Буквы математических гарнитур служат в формулах переменными, поэтому в шрифтах отсутствуют лигатуры и кернинг букв, но может быть кернинг переменных с символами операторов, отношений и т. д. Так как жирным прямым шрифтом принято обозначать векторы, а рубленый шрифт часто используется для обозначения тензоров, команды `\mathbf` и `\mathsf` должны печатать *переменные*, поэтому параметры mathbf и mathsf лучше установить в значение sym. То же рекомендуется сделать для остальных алфавитов. Исключением может быть прямой шрифт `\mathrm`, которым набираются названия функций, в которых использование лигатур уместно, сравните:

```
\unimathsetup{mathrm=sym}  $\operatorname{fit} y(x)$   ⇝   fit y(x)
\unimathsetup{mathrm=text} $\operatorname{fit} y(x)$   ⇝   fit y(x).
```

Декларация `\setoperatorfont{`*алфавит*`}` позволяет установить для печати функций любой другой алфавит. Например, после объявления `\setoperatorfont\mathsf`, получим `$\sin x$` ⇝ $\sin x$.

Использование алфавитов математической гарнитуры не создаст проблем с вводом в формулу *текста*, набранного жирным, рубленым или любым другим шрифтом. Команды `\textbf`, `\textsf` и т. д., или `\mathtextbf`, `\mathtextsf` и т. д. всегда позволят это сделать.

Как видно из табл. 12.9, большинство алфавитов представлено в OpenType шрифтах нормальными и жирными вариантами. В то же время OpenType технологии легко подключают жирные шрифты. Возникает вопрос: каким шрифтом печатать жирные буквы? Пакет unicode-math решает его следующим образом: декларации `\boldmath` и `\mathversion{bold}`, а также команда `\boldsymbol{•}` используют жирный шрифт, тогда как команды `\mathbf`, `\symbf` и `\sym•bf...` оперируют жирными вариантами алфавитов. Работая с пакетом unicode-math, командой `\bm{•}` лучше не пользоваться, так как она часто дает ошибки компиляции.[11] Это, пожалуй, единственная проблема верстки формул, набранных в стандартном LaTeX. При ее появлении замените

[11] Будем надеяться, что в ближайшем будущем пакеты bm и unicode-math будут согласованы, и команды `\bm` можно будет использовать наряду с `\boldsymbol`.

команду \bm на \boldsymbol, а если это не поможет, исключите из аргумента \boldsymbol команды \math... и \sym..., заменив их самими жирными символами или командами \sym•bf....

Пакет unicode-math дает возможность легко вводить дополнительные алфавиты и заменять существующие. Для этого определена декларация

\setmathfontface{*алфавит*}{*шрифт*}{*настройки*} .

Ее первым аргументом может быть существующая команда, например

\setmathfontface\mathtt{Anonumous Pro}}[Scale=...],

или новая команда для нового алфавита:

\setmathfontface\Garamond{EBGaramond-Italic}[Scale=MatchLowercase],
$\Garamond{ *abcdABCD* }$.

Если к настройкам шрифта добавить параметр version, алфавит будет добавлен (заменен) только к определенной версии шрифтов.

Декларации \setmathfontface должны находиться в преамбуле. В создаваемых алфавитах доступны только латинские буквы и цифры с ASCII кодами, дифференциал частных производных ∂ (U+2202), оператор градиента ∇ (U+2207), заглавные (U+391-a9) и строчные (U+3b1-c9) греческие буквы и их варианты ϑ (U+3d1), ϕ (U+3d5), ϖ (U+3d6), $\varkappa$ (U+3f0), ϱ (U+3f1) и ϵ (U+3f5).

Настройка гарнитур

Для работы с OpenType шрифтами unicode-math использует пакет fontspec, поэтому настройки текстовых и математических гарнитур имеют одинаковый синтаксис и много общих параметров типа Scale, SizeFeatures и т. д. В то же время у математических гарнитур есть и свои специфические настройки, перечисленные в табл. 12.7, которые устанавливает ранее упоминавшаяся декларация

\setmathfont{*гарнитура*}[*настройки*] .

Рассмотрим их вкратце.

Прежде всего, отметим несколько важных обстоятельств. Во-первых, декларации \setmathfont являются глобальными, их действия не ограничены группой. Во-вторых, помимо основной гарнитуры можно настроить любое количество других гарнитур, или версий. Версии можно создавать только после *явной* установки основной гарнитуры, которой по умолчанию присваивается версия normal. Настройки, сделанные для основной гарнитуры, сбрасываются, когда активируется новая гарнитура, но настройки версий остаются неизменными.

Очень хорошо проработанная гарнитура STIX в настоящее время не имеет жирного шрифта, который есть в близких по начертанию шрифтах XITS. Установка

```
\setmathfont{STIX Two Math}[настройки]
\setmathfont{XITSMath-Bold}[version=bold, настройки]
```

определит гарнитуру STIX Two Math в качестве основного шрифта (версия normal) и подключит шрифт XITSMath-Bold в качестве жирного. Его будут активировать декларации `\boldmath`, `\mathversion` и команда `\boldsymbol{•}`. Подключение выполнит параметр version, предназначенный для создания версий математических шрифтов. Еще один пример, приведенный в начале данного раздела, демонстрирует использование параметра version для создания дополнительной гарнитуры «Garamond».

Активирует версии шрифтов декларация `\mathversion{версия}`, которая не является глобальной. Чтобы вернуться к основной гарнитуре, поработав со шрифтами другой версии, достаточно выйти из группы, или применить декларацию `\mathversion{normal}`.

При верстке гарнитурами, разработанными в единственном размере, индексы печатаются уменьшенным шрифтом, но как уже говорилось, масштабирование шрифта ухудшает его качество. Если в гарнитуре есть мелкие шрифты, их можно подключить, используя параметр script-font для первых индексов и sscript-font для вторых, например:

```
\setmathfont{Minion Math}[
        script-font      = MinionMath-Capt,
        sscript-font     = MinionMath-Tiny,
        script-features  = {Style = MathScript, настройки},
        sscript-features = {Style = MathScriptScript, настройки} ]
```

Настройки шрифтов script-features и sscript-features должны содержать параметр Style, а если они не используются, указанные атрибуты присваиваются шрифтам автоматически.

Параметр range позволяет заменить группу символов гарнитуры другими символами. Для этого сначала декларацией `\setmathfont`, в которой отсутствует параметр range, устанавливают основную гарнитуру, а затем декларациями `\setmathfont` с параметром range указывают, какие символы следует подставить из этих гарнитур. Таким образом, составляется «рабочая» гарнитура, используемая в верстке. Она может содержать глифы разных шрифтов, но даже если символы подставляются из той же гарнитуры, чтобы сформировать рабочую гарнитуру, нужны две декларации.

Группу можно задать диапазоном шестнадцатеричных кодов, range= {"27D0-"27EB, "27FF}, списком команд, range= {\int,\sum}, или категорией символов, range=\mathop (см. табл. 12.5 и прил. В). Если в группу входит несколько диапазонов, их разделяют запятыми и заключают в фигурные скобки.

Для групп-алфавитов можно использовать команды \sym..., например range= {\symcal,\symbfcal}, или аббревиатуры, приведенные в табл. 12.8. Алфавиты разделяются на подгруппы из цифр и букв: num, Latin, latin, Greek, greek. Строчные и прописные, латинские и греческие буквы выделяются в отдельные подгруппы, обозначения прописных букв начинаются с заглавной буквы. Диапазон подгруппы формируют аббревиатура алфавита и обозначение подгруппы, разделенные слэшем. Покажем для примера, как указывается алфавит прямых букв, range=up, и его части содержащие греческие буквы, range=up/{greek,Greek}, и строчные латинские буквы, range=up/latin.

Если в гарнитуру подставляются символы другой гарнитуры, достаточно указать диапазон замены. Например, так подключается рукописный алфавит шрифта XITS:

```
\setmathfont{XITS Math}[range={\symscr,\symbfscr}]
```

Когда алфавит, или его подгруппа, заменяется другим алфавитом того же шрифта, мнемоника меняется. В этом случае необходимо указать диапазон заменяемых символов и заменяющий алфавит, соединив их «знаком замены» ->. Например, так можно сделать курсивные строчные греческие буквы прямыми:

range={it/greek->up,bfit/greek->bfup}.

Такая замена уже использовалась в настройках шрифтов в разд. 12.3.

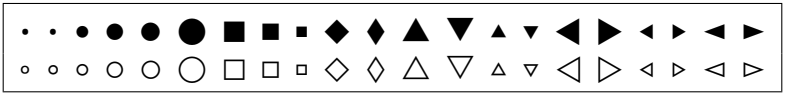
Ввиду экспериментального статуса пакета unicode-math при использовании версий шрифтов могут возникать проблемы, связанные с заменой символов основной гарнитуры. Чтобы избежать их, нужно соблюдать определенный порядок настройки шрифтов. Сначала задают основную гарнитуру, затем версии шрифтов и в конце настраивают параметр range основной гарнитуры. Как показывает опыт, объявление версии шрифтов сбрасывает сделанные ранее замены.

Символы

OpenType шрифты вмещают большое количество символов, которые удобно вводить в формулы в «естественном» виде, минуя команды

LaTeX. Небольшая часть обширного набора геометрических символов для примера показана в табл. 12.10.

Таблица 12.10
Геометрические символы OpenType шрифтов



Под действием команд `\sym...` или `\math...` вид символов не меняется. Команды `\boldsymbol` делают их жирными, только если подключена жирная версия шрифта, так как в основной гарнитуре жирные символы есть лишь у алфавитов.

Работу с символами, определенными в кодировке unicode неоднозначно, можно настроить. К таким символам относится двоеточие, имеющее разные отбивки: знак пунктуации `U+3A` отделяется пробелом справа, `1: 1`, а оператор отношения `U+2236` — симметрично с обеих сторон, `1 : 1`. С клавиатуры вводится знак пунктуации, но чтобы в формулах он воспринимался как оператор, unicode-math вводит параметр настройки colon и по умолчанию присваивает ему значение TeX, если же задать значение literal, двоеточие станет знаком пунктуации. Нужное двоеточие можно также напечатать командами `\symliteral` и `\symnormal`.

Слэш представлен в unicode четырьмя вариантами: слэш / (`U+2f`), дробная черта ⁄ (`U+2044`), знак деления ∕ (`U+2215`) и большая дробная черта ⧸ (`U+29f8`). Только первый из них определен в LaTeX как команда `\slash` или символ «/», вводимый с клавиатуры. Обратный слэш также имеет четыре варианта, три из которых определены в LaTeX: обратный слэш \ (`U+5C`) — команда `\backslash`, символ ∖ (`U+2216`) — команда `\smallsetminus`, обратная дробная черта ⧵ (`U+29f5`) — команда `\setminus` и большая обратная дробная черта ⟍ (`U+29f9`). По умолчанию в LaTeX символы `U+2f` и `U+5c` служат разделителями переменного размера в конструкциях `\big•` и `\left•...s\right•`. Однако в OpenType слэш, способный менять свой размер, не задан жестко, поэтому пакет unicode-math позволяет выбрать его с помощью параметра `slash-delimiter`, который может принимать значения ascii — символ `U+2a` (по умолчанию), frac — символ `U+2044` или div — символ `U+2215`. Второй большой разделитель определен однозначно, им всегда выступает обратный слэш `U+5c`.

Индексы и дроби

Кодировка unicode содержит небольшое количество символов нижних и верхних индексов и дробей. Их можно использовать в наборе, например печатать $(\alpha^2)^n=\alpha^{2n}$ вместо `(\alpha^2)^n=\alpha^{2n}`.

Таблица 12.11
Индексы и дроби OpenType шрифтов

0 1 2 3 4 5 6 7 8 9 + − = ( ) i n n h j r w y
½ ⁰⁄₃ ⅓ ⅔ ¼ ¾ ⅕ ⅖ ⅗ ⅘ ⅙ ⅚ ⅐ ⅛ ⅜ ⅝ ⅞ ⅑ ⅒
0 1 2 3 4 5 6 7 8 9 + − = ( ) a e i o r u v x β γ ϱ φ χ

Символы верхних и нижних индексов OpenType шрифтов приведены, соответственно, в верхней и нижней строке табл. 12.11. Среди них есть цифры, знаки сложения, вычитания, равенства, круглые скобки и некоторые буквы.

Символы дробей после компиляции превращаются в обычные дроби, например:

$$\textbf{1}\!\!\textbf{/}\!\!\textbf{2}\,x+\textbf{1}\!\!\textbf{/}\!\!\textbf{2}\,y=\textbf{1}\!\!\textbf{/}\!\!\textbf{2}\,(x+y) \rightsquigarrow \tfrac{1}{2}x+\tfrac{1}{2}y=\tfrac{1}{2}(x+y).$$

Размер дробей зависит от значения параметра active-frac. Применяемая по умолчанию настройка active-frac=small, верстает дроби командой `\tfac`, а если поменять ее на active-frac=normalsize, будет использоваться команда `\frac`, регулирующая размер дробей в зависимости от контекста.

Символы с отрицанием

Как говорилось на с. 88, команда `\not{•}` применяется в LaTeX для построения символов с отрицанием. Пакет unicode-math поддерживает ее в полной мере и, более того, позволяет определить для печати символа с отрицанием собственную команду

`\NewNegationCommand{`*символ*`}{`*команда*`}` .

После определения `\NewNegationCommand{≡}\nequiv` команды `\nequiv` и `\not\equiv` и будут давать одинаковый результат $\rightsquigarrow \not\equiv$. Повторное определение уже имеющейся команды вызовет ошибку. Для ее изменения предназначена декларация

`\RenewNegationCommand{`*символ*`}{`*команда*`}` .

К сожалению, ввиду экспериментального статуса пакета unicode-math, данная команда пока ничего не делает. Надо надеяться, что со временем эта оплошность будет устранена.

Резюме

Поводя итоги, опишем настройки математических шрифтов, обеспечивающие оптимальные ресурсы верстки формул. За основу возьмем шрифты STIX2 и XITS, символы которых имеют одинаковую контрастность, хорошо сочетающуюся со шрифтами Times New Roman:

```
\usepackage[mathbf=sym, mathsf=sym]{unicode-math}
\setmathfont{STIX Two Math}
\setmathfont{XITSMath-Bold}[version=bold]
\setmathfont{STIX Two Math}[range={it/greek->up,bfit/greek->bfup}]
\setmathfont{XITS Math}[range={\symscr,\symbfscr}]
```

При загрузке пакета параметры mathbf и mathsf устанавливаются в значение sym, чтобы векторы и тензоры, введенные командами `\mathbf` `\mathsf`, оптимально сочетались с другими символами. Гарнитура STIX Two Math активируется с настройками TeX (см. табл. 12.8), используемыми по умолчанию. Затем подключается жирный шрифт XITSMath-Bold. Стиль TeX в целом соответствует традициям российской полиграфии, лишь курсивные строчные греческие буквы нужно заменить прямыми, что и делает следующая декларация. Затем рукописный алфавит шрифта XITS Math добавляется к каллиграфическому, имеющемуся в гарнитуре STIX Two Math. Эта настройка позволит вводить буквы каллиграфического алфавита с помощью команды `\mathcal` или в виде символов, а доступ к рукописному даст команда `\mathscr`. Как отмечалось ранее, чтобы объявление жирного шрифта не сбросило дополнительные настройки символов основной гарнитуры, нужно соблюдать указанный порядок деклараций.

В англоязычных документах формулы обычно верстаются в стиле TeX, поэтому «выпрямлять» греческие буквы не нужно, а остальные настройки не меняются.

Заключение

Концепция программной верстки начала развиваться, когда графические возможности мониторов были крайне ограничены. Рост производительности компьютеров и появление дисплеев с высоким разрешением породили концепцию визуальной верстки WYSIWYG — What You See Is What You Get, лежащую в основе офисных редакторов типа MSWord. Каждая концепция имеет свои преимущества и недостатки.

Преимуществом визуальной верстки является наглядность, значительно облегчающая редактирование. Она достигается ценой сильной связи документа с конфигурацией программного обеспечения, установленного на компьютере автора, поэтому воспроизведение на другом компьютере или копирование из одного документа в другой часто вызывает трудности: изменяются шрифты, рисунки «уплывают» из отведенного им места, искажаются ячейки в таблицах, формулы зачастую становятся нередактируемыми. Формат документов слабо стандартизован, и при изменении стиля верстки большую часть работы придется делать заново. Перечисленные недостатки не мешают редактированию офисной документации и простых текстов, поэтому WYSIWYG редакторы стали привычным инструментом, широко представленным на рынке программного обеспечения. Коммерциализация препятствует стандартизации визуальной верстки, поскольку в целях извлечения прибыли в обновления «офисных» программ целенаправленно закладывается какая-либо несовместимость с предыдущими версиями.

Преимуществами программной верстки средствами LaTeX являются воспроизводимость документа практически на любом компьютере, высокая совместимость частей разных документов, гибкость настроек параметров верстки и очень простое изменение ее стиля, практически не требующее переделки самого документа. Добавим к этому высокое качество верстки формул и удобство работы с перекрестными ссылками и библиографией. Недостатком является слабая поддержка кодировки unicode в нынешнем стандарте LaTeX 2_ε, вынуждающая представлять символы командами, сильно загромождающими формулы. Эту проблему легко решают современные компиляторы XeTeX и LuaTeX, верстающие тексты, набранные в этой кодировке. Разработанные для них форматы XeLaTeX и LuaLaTeX вместе с пакетами fontspec [86] и unicode-math [87] позволяют избавить программу от команд-символов, однако здоровый консерватизм пока сдерживает издателей от внедрения их в свою практику.

Еще один недостаток, связанный со значительным отличием рукописи от сверстанного документа, устраняет графическая оболочка LyX, основанная на концепции WYSIWYM — What You See Is What You Mean, адаптирующей технологии визуальной верстки для LaTeX. Свободно распространяемая программа LyX реализована для всех операционных систем и активно развивается. Она использует собственный формат рукописи, позволяющий заменить команды и конструкции LaTeX их графическим представлением, т. е. вместо команды `\section` в окне редактора отражается название раздела в соответствующем шрифтовом оформлении, а формулы имеют графический вид и редактируются с помощью различных меню. Логика программной верстки сохраняется, но набор рукописи существенно облегчается и ускоряется.

Рукопись LaTeX можно импортировать в LyX, а по окончании редактирования экспортировать обратно. Так как синтаксис LaTeX очень гибок, код, конечно, не будет оптимальным, более того, он может содержать нестандартные решения, поскольку экспорт основан не на оптимальных, а на универсальных алгоритмах, но это — плата за квазивизуальную верстку.

Заинтересованный читатель найдет LyX на сайте `www.lyx.org` или в пакетах своей операционной системы. Обширная документация LyX имеет собственную Wiki-страницу по адресу `wiki.lyx.org`. Следует, однако, помнить, что конечная цель автора — генерация кода LaTeX, пригодного для представления в издательство, поэтому для осознанной и успешной работе в LyX нужно сначала освоить LaTeX, хотя бы в объеме первой части этой книги.

Следующим шагом в развитии LaTeX является освоение облачных технологий. В сети уже имеется значительное число бесплатных и коммерческих online-редакторов, позволяющих работать с рукописью на любом компьютере, подключенном к сети Интернет. Все необходимое для верстки документа хранится в облаке, что очень удобно при коллективном редактировании манускрипта несколькими авторами. Сохраняемая история редактирования помогает определить исправления, вносимые различными авторами. Для комфортной работы с online-редакторами необходим высокоскоростной интернет.

На сегодняшний день наиболее развит, популярен и удобен редактор Overleaf. Чтобы верстать документы, достаточно зарегистрироваться на его сайте по адресу `https://www.overleaf.com`. Бесплатное пользование обеспечивает полноценную работу и ее поддержу обширной и хорошо структурированной документацией по LaTeX и самому Overleaf.

Редактируемая рукопись представляет собой обычный код LaTeX, в него не добавляются никакие дополнительные команды и ресурсы. Текст рукописи можно представить в компактном и удобном виде, в котором команды, встречающиеся наиболее часто, типа `\section`, `\label`, `\ref`, `\cite` и т. п. отображаются в виде заголовков разделов, имен меток и ярлыков, а формулы представляются в компилированном виде. Нажатие мышью «разворачивает» компактный объект в команды, а после редактирования он вновь «сворачивается». После компиляции в правой части окна, аналогично TeXstudio, показывается сверстанный документ. Между окнами редактирования и просмотра имеется связь, позволяющая удобно переходить в нужную часть документа.

Используя Overleaf бесплатно, автор может предоставить доступ к редактированию документа еще одному пользователю. Оплаченная работа добавляет возможность пользоваться графическим меню для ввода символов, право подключения к коллективной работе до десяти пользователей, а также приоритет на online-компиляцию перед бесплатными пользователями.

Комплексные исследования, являющиеся основой современной науки, создание коллабораций для решения сложных задач и широкое международное сотрудничество научных коллективов делают облачные технологии LaTeX чрезвычайно востребованным инструментом, помогающим готовить к опубликованию результаты совместной работы.

Приложения

Приложение А

TEXнические детали

А.1. Компиляция в TEXstudio

Графическая оболочка служит для набора рукописи и организует цепочку ее компиляции, запуская другие программы. В программе TeXstudio средства компиляции находятся в меню «Tools». Его строка «Build & View» и иконка панели инструментов (см. рис 1.1) запускают полную сборку документа, включающую создание оглавления, библиографии и предметного указателя, в которой участвуют несколько компиляторов. По окончании компиляции открывается окно просмотра pdf-документов или программа, показывающая готовый документ, если он имеет другой формат. Строка «Compile» и иконка запускают используемый по умолчанию компилятор LATEX и, при необходимости, постпроцессор. Строки «Bibliography», «Glossary» и «Index» запускают программы, формирующие библиографию, глоссарий и предметный указатель. Различные варианты компиляции документа настраиваются в меню «Options ⇒ Configure TeXstudio...⇒ Build». Настройки компиляторов содержит меню «Options ⇒ Configure TeXstudio...⇒ Commands». В нем можно посмотреть и отредактировать список ключей, с которыми запускаются программы. В меню «Tools ⇒ Commands» можно по отдельности запустить каждую из программ.

Если компиляция выявила ошибки, TEXstudio открывает окно для их анализа и устранения. По умолчанию в нем активируется закладка

Messages или Log, показывающая сообщения компиляторов, извлекаемые оболочкой из генерируемых программами log-файлов. Полный текст log-файлов доступен в закладке Issues.

В сложных случаях, когда оболочка не справляется с обработкой результатов компиляции, ее проще провести вручную. Рассмотрим, как это делается.

А.2. Компиляция вручную

Программы, участвующие в создании документа, графического интерфейса не имеют. Их нужно запускать из командной строки в консоли. В Unix-подобных системах открыть консоль не проблема, однако многие пользователи ОС Windows даже не подозревают о ее существовании, поэтому дадим краткую инструкцию.

Компиляцию нужно проводить в рабочей папке, в которой находится документ. Откройте проводник и перейдите в рабочую папку. После этого в окне проводника нажмите «Shift» и правую кнопку мыши. В появившемся меню выберите строку «Открыть окно команд», или что-то подобное.[1] В папке, в которой вы находитесь, откроется консоль. Чтобы настроить ее, нажмите на строку заголовка окна консоли правой клавишей мыши и выберите строку «Свойства» в выпавшем меню. Установите комфортный для глаз шрифт и ширину окна не менее 100-120 символов, чтобы в его строке полностью умещались сообщения, генерируемые запускаемыми программами. Нажав клавишу «OK» или «Сохранить», можно приступать к работе.

Опишем наиболее сложный процесс интерактивной работы компилятора LaTeX, взяв для примера программу pdflatex.

Чтобы выполнить компиляцию, наберите имя программы (ее расширение •.exe в Windows можно опустить) и введите имя компилируемого документа, например,

```
pdflatex sample.tex.
```

Начав набор имени, попробуйте нажать клавишу «Tab» клавиатуры, инициирующую автоматическое завершение его ввода. Программа консоли (оболочка) просматривает имена файлов в рабочей папке и подставляет их в командную строку. Если несколько файлов имеют сходные имена, на консоль выводится их общая часть, а при повторном

[1] Название строки зависит от версии ОС Windows.

нажатии «Tab» — подсказка в виде списка имен. Закончив ввод имени, нажмите клавишу «Enter» и запустите компиляцию.

Вручную компиляцию лучше запускать без ключей. При необходимости их можно добавить в командную строку, перечислив через пробел без запятых между именем программы и именем файла. Программы дистрибутивов TEX, запускаемые с ключом `--help`, печатают список ключей, кратко комментируя их назначение. Вот какую справку выдает компилятор библиографии:

```
bibtex --help

Usage: bibtex [OPTION]... AUXFILE[.aux]

  Write bibliography for entries in AUXFILE to AUXFILE.bbl,
  along with a log file AUXFILE.blg.

-min-crossrefs=NUMBER include item after NUMBER cross-refs;
                      default 2
-terse                do not print progress reports
-help                 display this help and exit
-version              output version information and exit
```

При работе программы выдают на консоль сообщения о выполняемых действиях, загружаемых файлах и обнаруженных ошибках. Компиляторы LATEX печатают сообщения обо всех загружаемых пакетах, шрифтах и других файлах, а обнаружив ошибки, останавливаются и печатают сообщения, начинающиеся восклицательным знаком. Они содержат описание ошибки, за которым следует номер и часть строки компилируемого файла, содержащей ошибку. Перед номером печатается буква «l» и точка:

```
! Undefined control sequence.
l.26 ...и обнаружив ошибку, \ErrorSample
?
```

В приведенном примере говорится, что команда `\ErrorSample` не определена. Введя букву «h» (и нажав «Enter»), можно получить более подробный комментарий к ошибке. При вводе «?» печатается справка о возможных дальнейших действиях. Полный список сообщений об ошибках и советы по их устранению приведены в книге [3].

Изучив ошибку, лучше сразу ее исправить, так как она может служить причиной следующих, после чего нужно нажать клавишу «Enter», чтобы продолжить компиляцию. Если ошибок много, введите букву «s», тогда компиляция продолжится без остановок и закончится генерацией файла с документом. Однако если ошибок слишком много, или одна из них не поддается обработке, компиляция может прерваться,

при этом программа не завершит работу и будет ждать дальнейших указаний. Тогда ее нужно остановить принудительно, введя букву «x». При такой аварийной остановке новый файл с документом не генерируется, а старый уничтожается.

После окончания компиляции подробный отчет о своей работе компиляторы TEX сохраняют в файле с расширением •.log, который можно просмотреть в программе Notepad или подобной ей.

Программы, формирующие библиографию и указатели, интерактивную компиляцию не поддерживают. Свою работу они также комментируют сообщениями, выводя их на консоль и сохраняя в файлах с расширениями •.blg и •.ilg.

А.3. Дистрибутив MiKTEX

Среди существующих дистрибутивов Mik!TeX наиболее удобен как в работе, так и в настройке. Он автоматически устанавливает недостающие пакеты, а для настройки использует программу MiKTEXConsole, рабочее окно которой показано на рис. А.1.

В левой части окна находится список меню, которые выполняют различные настройки. Меню «Overview» содержит общую информацию о дистрибутиве. Меню «Update» позволяет узнать список пакетов, обновленных на сайте MiKTEX, и установить их. Меню «Packages» и «Documentation» предназначены для установки пакетов и их документации вручную. В меню «Diagnose» можно посмотреть сообщения об ошибках в работе MiKTEX и сводку информации о дистрибутиве. Меню «Cleanup» предназначено для удаления файлов из папок дистрибутива перед удалением самого MiKTEX.

Меню «Settings» содержит несколько панелей. В панели «General» можно выбрать метод автоматической установки пакетов с уведомлением или без него, или отменить автоматическую установку. Панель «Formats» позволяет удалить или добавить форматные файлы различных компиляторов, которых помимо pdflatex и lualatex насчитывается более сорока, а панель «Languages» — включает или отключает поддержку переносов для различных языков. Панель «Directories» настраивает папки, содержащие программы и ресурсы, используемые при компиляции. В папке «...\Roaming\MiKTeX» (.../texmfs/config) находятся конфигурационные файлы, в папку «...\Programs\MiKTeX» (.../texmfs/install) устанавливаются пакеты, а в папку «...\Local\MiKTeX» (.../texmfs/data) направляются ресурсы, генерируемые в процессе

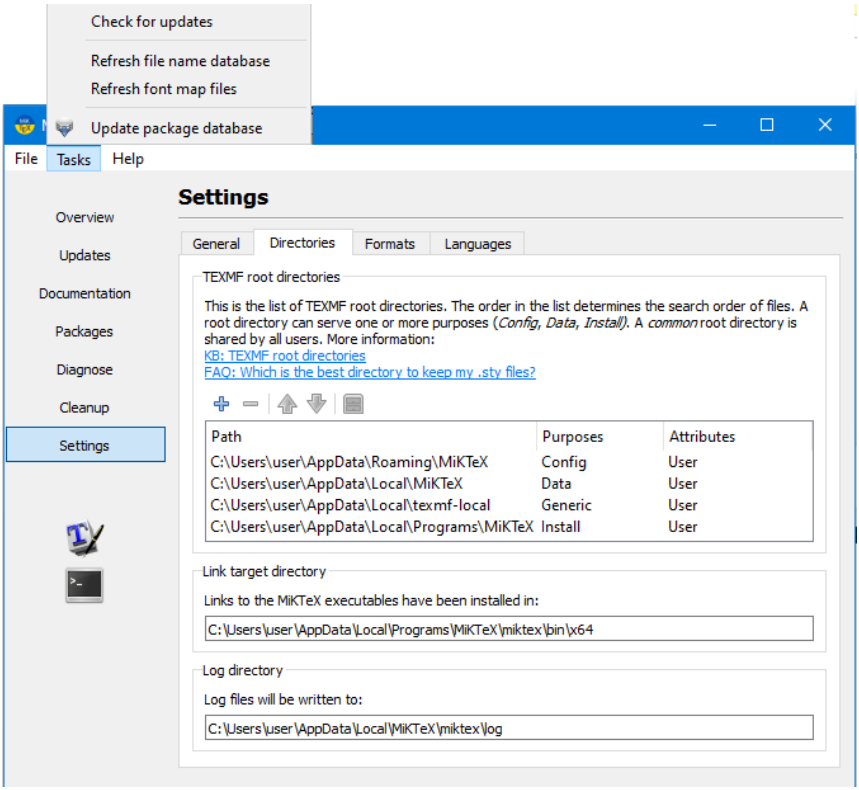


Рис. А.1. Рабочее окно программы MiKTEXConsole

компиляции и используемые в дальнейшей работе, например метрики шрифтов или растровые шрифты. Папки с именами, заключенными в кавычки, создаются в ОС Windows, а в скобках приведены названия папок в ОС Linux. В дистрибутиве TEXLive аналогичные папки называются .../texmf-cfg, .../texmf-var - и .../texmf-dist. Не следует размещать в перечисленных системных папках ресурсы, добавляемые самостоятельно, например классы документов и пакеты, отсутствующие в MiKTEX. Для них нужно создать отдельную (локальную) папку и внести ее в список обслуживаемых папок, нажав символ «+», над окошком со списком. Файлы, находящиеся в локальных папках, не уничтожаются при удалении MiKTEX. В примере на рис. А.1 локальной папкой является «...\Local\texmf-local».

Дистрибутивы TEX содержат большое количество файлов и чтобы ускорить их поиск и загрузку, информация об их расположении заносится в специальную базу данных. Если файл не находится в рабочей папке и информация о нем отсутствует в базе, он не будет найден. После установки пакетов средствами MiKTEX база обновляется автоматически, если же понадобится добавить какие-либо дополнительные ресурсы, поместите их в локальную папку и обновите базу вручную, выбрав строчку «Refresh file name database» в меню «Tasks».

Поиск ресурсов, обеспечивающих компиляцию, начинается с рабочей папки, за ней следуют локальные папки пользователя, и в последнюю очередь просматриваются системные папки.

Для поиска файлов компиляторы вызывают программу kpsewhich, которой можно пользоваться, чтобы проверить наличие того или иного файла. Запущенная в консоли, она напечатает полное имя файла, включая путь к нему, например:

```
kpsewhich article.cls
C:/Users/user/AppData/Local/Programs/MiKTeX/tex/latex/base/article.cls
```

Чтобы ресурс был доступен компилятору, недостаточно только разместить его в системной или локальной папке. Папки дистрибутивов TEX имеют определенную структуру и искомые файлы должны занимать в них определенные места.

Системные папки организованы в соответствии с назначением ресурсов. Их принято называть общим именем `/texmf-•`, происходящим от программ TEX и METAFONT, созданных Д. Кнутом. Давайте рассмотрим структуру папок с ресурсами компиляторов .../texmf-dist, ...\Local\MiKTeX и .../install, обозначив их /texmf. Находящиеся в них папки называются именами программ или ресурсов, например /texmf/tex или /texmf/fonts. На рис. А.2 показана структура стандартной папки, вернее ее часть, содержащая папки с пакетами, стилевыми файлами, шрифтами и т. п., упоминавшимися в книге. Назначение папок следует из их названия. Мелким шрифтом в скобках указаны расширения преимущественно хранящихся в них файлов. Жирным шрифтом выделены папки, на основе которых рекомендуется построить структуру локальной папки для хранения собственных ресурсов.

В папке /bibtex хранятся ресурсы верстки библиографии: базы данных (•.bib), стилевые файлы (•.bst) и кодировки (•.csf), используемые для сортировки списков литературы. Библиографические базы, помещенные в папку /bibtex/bib, можно использовать при компиляции многих документов, не перенося их для этого в рабочие папки.

Рис. А.2. Структура каталога texmf

Папка /doc содержит документацию. Некоторые пакеты размещают ее там непосредственно при собственной установке, но чаще всего она выносится в отдельные пакеты, которые можно установить в упоминавшемся ранее меню «Documentation». Отметим, что подавляющее большинство пакетов LaTeX имеет руководство пользователя.

В папке /fonts находятся ресурсы шрифтов. Папки /fonts/tfm и /fonts/mf занимают метрики шрифтов и ресурсы программы METAFONT, необходимые для генерации растровых шрифтов. Ресурсы PostScript шрифтов занимают несколько папок. Сами шрифты устанавливаются в папку /fonts/type1, а их метрики, кодировки и карты в папки /fonts/afm, /fonts/enc и /fonts/map. OpenType и TrueType шрифты размещаются в одноименных папках и смешивать их нельзя.

Папку /scripts занимают скрипты, выполняющие различные операции. Как правило, для их исполнения требуется, чтобы в системе были установлены соответствующие компиляторы-интерпретаторы, не входящие в дистрибутивы TeX. Например, скрипты с расширением •.pl исполняет программа Perl, малодоступная в ОС Windows.

Подавляющее число пакетов устанавливается в папку /tex. Она содержит папки /tex/latex и /tex/lualatex с ресурсами компиляторов LaTeX и LuaLaTeX, при этом последний использует ресурсы обеих папок, а первый — только своей. В папке /tex/lualatex/xindex находятся системные настройки программы xindex. Ресурсы программы xindy хранятся в ее собственной папке /xindy.

Локальные настройки программ xindy и xindex следует размещать в папках /texmf-local/xindy и /texmf-local/tex/lualatex/xindex, шрифты устанавливать в папки /texmf-local/fonts/opentype и /texmf-local/fonts/truetype, а их описания в папку /texmf-local/tex/latex/fontspec. Классы документов, пакеты и стилевые файлы LaTeX должны находиться в папке /texmf-local/tex/latex.

В завершение отметим, что дистрибутив MiKTeX оперативно обновляет пакеты, отслеживая их свежие версии в центральном TeX-архиве на сайте `www.ctan.org`. Пакеты и их документация находятся на этом сайте по адресам `www.ctan.org/pkg/`*имя пакета*.

A.4. Программа otfinfo

Программа otfinfo выводит на консоль информацию о параметрах OpenType шрифтов. При ее запуске нужно указать ключ и имя файла:

`otfinfo` *ключ имя файла*.

Запуская ее в папке, содержащей шрифт, достаточно указать *имя файла* с расширением, в иных случаях *имя* должно содержать путь к файлу. Программа запускается с *одним* из ключей, печатающих разную информацию:

-h, --help — полный список ключей;

-i, --info — наименование гарнитуры, имя шрифта, версия, разработчик, производитель и лицензионные права;

-a, --family — наименование гарнитуры;

-p, --postscript-name — имя шрифта;

-v, --font-version — номер версии шрифта;

-s, --scripts — поддерживаемые алфавиты;

-f, --features — коллекции альтернативных начертаний;

-z, --optical-size — оптические размеры;

-g, --glyphs — список глифов;

-t, --tables — список встроенных таблиц;

-T, --dump-table *таблица* — распечатка встроенной таблицы;

-q, --quiet — не печатать сообщения об ошибках;

--version — версия программы.

Ключ -q можно использовать вместе с другими ключами. Ключ -i выводит несколько наименований шрифта: его PostScript имя, название семейства (Family), полное имя (Full name), наименование гарнитуры (Preferred family), идентификатор (Unique ID). Пакет fontspec идентифицирует шрифт по названию гарнитуры. Если поле Preferred family отсутствует в выводе, для идентификации нужно использовать название семейства (Family).

A.5. Программа JabRef

Программа JabRef, обслуживающая базы bibtex, доступна по адресу `https://www.jabref.org`. Она способна аккумулировать данные многих интернет-баз, специализирующихся на различных отраслях науки, и может конвертировать в формат bibitex более пятнадцати других форматов библиографических источников.

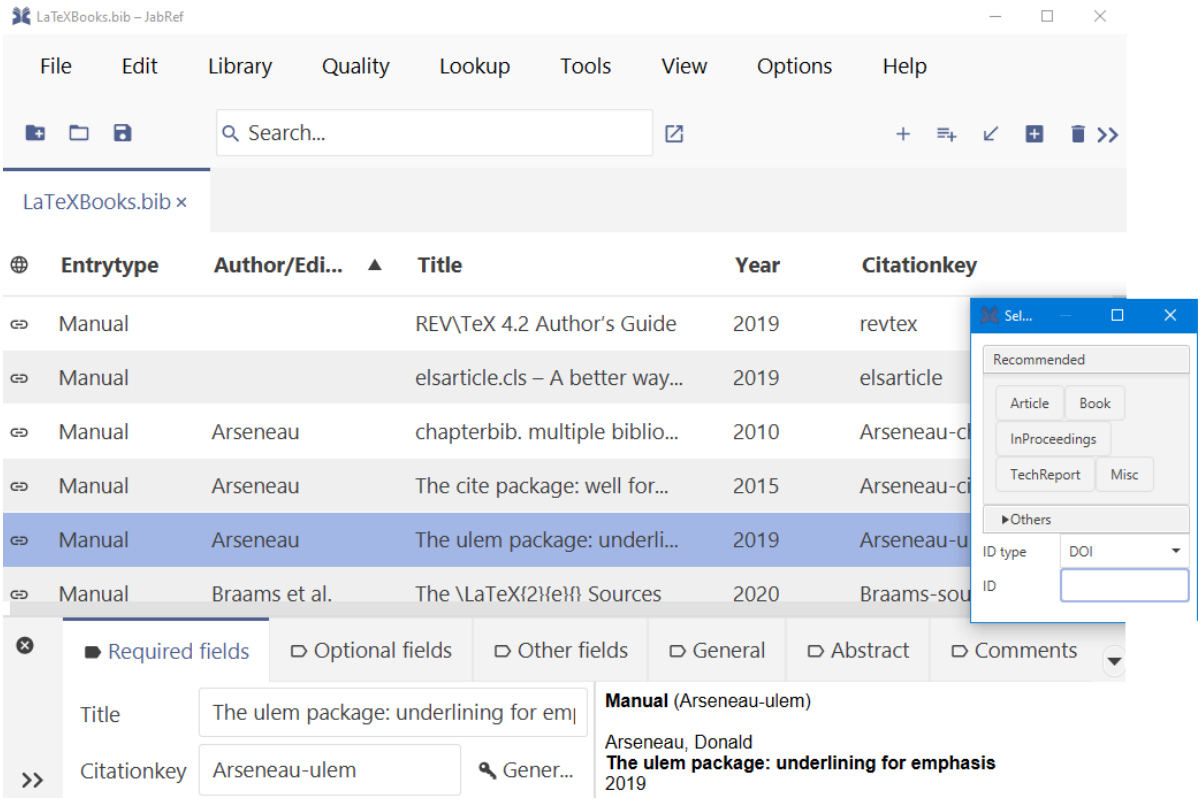


Рис. А.3. Рабочее окно программы JabRef

Рабочее окно программы, показанное на рис. А.3, делится на несколько частей. Его верхняя часть содержит основные меню и инструментальную панель с поисковой строкой, позволяющей быстро перейти к нужной записи в большой базе. Основную часть занимает список записей, который можно сортировать по различным полям, набор которых настраивается. При двукратном нажатии на строку списка в нижней части окна открывается набор панелей редактирования, показывающих содержимое записи и позволяющих внести изменения в ее поля. Редактирование с их помощью ограждает от случайного нарушения структуры записей и базы в целом.

Новую запись создает всплывающее окно, которое открывает комбинация клавиш «Ctrl+N» или иконка , находящаяся справа от поисковой строки. Основные типы записей представлены в нем «клавишами», а другие доступны в меню «▸Others». Запись можно импортировать из интернета, введя идентификатор его ресурса в строку ввода «ID». По умолчанию программа обращается к базе DOI, но меню «DOI ▾» содержит множество других доступных баз. Его название изменяется в зависимости от его последнего состояния. Оно соответствует названию базы, к которой обращались перед закрытием меню, например, после обращения в к базе ArXiv оно будет называться «ArXiv ▾». Если ресурс неизвестен, откройте меню, выберите в его списке «Title» и введите в строку ввода заголовок статьи или название книги. JabRef попытается найти их самостоятельно.

Приведенных сведений достаточно, чтобы начать работу с программой. Ее интерфейс прост и понятен, поэтому дальнейшие действия можно освоить самостоятельно.

А.6. Создание и обработка иллюстраций

В составе ОС Windows поставляется PostScript-драйвер, однако он не устанавливается по умолчанию. При работе с ним возникает масса проблем, некоторые из которых неустранимы, поэтому мы рекомендуем пользоваться свободно распространяемым драйвером корпорации Hewlett-Packard. Зайдите на сайт

`https://support.hp.com/us-en/drivers/printers`

и введите в меню поиска «HP Universal Print Driver Series for Windows», затем выберите «Driver-Universal Print Driver», загрузите драйвер «HP Universal Print Driver for Windows **PostScript**» и устано-

вите его. Для этого в настройках ОС Windows[2] «Принтеры и сканеры ⇒ Добавить принтер или сканер» выберите «Установка принтера ⇒ Добавить локальный или сетевой принтер с параметрами, заданными вручную». В меню «Используйте существующий порт» выберите «FILE:(Печать в файл)», установите принтер и в конце установки запретите общий доступ к нему. Затем принтер нужно настроить, чтобы он генерировал корректный PostScript-код. Найдите его в меню «Принтеры и сканеры» и зайдите в меню «Управление ⇒ Настройки печати». В появившемся окне выберите закладку Прочие. В меню «Параметры документа ⇒ Параметры PostScript ⇒ ⇒ Параметры вывода PostScript» выберите Инкапулированный PostScript (EPS) и нажмите клавишу «**Применить**», а затем клавишу «OK». При нажатии только клавиши «OK» настройка не сохранится! Отметим, что меню печати ОС Windows позволяет настроить некоторые параметры принтера непосредственно перед печатью, но в нем клавиша «Применить» недоступна, поэтому установить правильный формат PostScript-вывода с его помощью не удастся.

После установки драйвера PostScript-рисунок можно создать путем печати из любого приложения. Создаваемому файлу ОС Windows установит расширение •.prn, которое нужно исправить на •.ps. «Напечатанный» рисунок будет окружен пустыми полями, так как его размеры задаст размер страницы, на которой он «печатается». Чтобы убрать поля, воспользуйтесь графическим редактором inkscape. Его можно установить, загрузив с сайта `https://inkscape.org/`.

В программе inkscape создайте новый документ и импортируйте в него PostScript-файл. Импортированный рисунок будет содержать несколько «слоев», сгруппированных вместе. Разгруппируйте их в меню «Объект ⇒ Разгруппировать» (Ctrl+Shift+G). Выберите объекты, составляющие рисунок, и сгруппируйте их вместе в меню «Объект ⇒ ⇒ Сгруппировать» (Ctrl+G). Создайте еще один новый документ, скопируйте в него сгруппированный рисунок и сохраните в формате •.svg, чтобы иметь возможность редактировать его, используя широкие возможности программы inkscape. После этого в меню «Файл ⇒ Сохранить копию...» сохраните рисунок в одном из форматов •.pdf, •.ps или •.png, поставив флажок «Использовать размер экспортируемого объекта».

В ОС Linux для генерации PostScript-рисунка специальных драйверов не требуется. Скорректировать BoundingBox позволяет программа ps2eps, запускаемая командной строкой:

[2] Нахождение описываемых далее меню зависит от версии ОС Windows.

```
ps2eps -B -g -f •.ps.
```

Исходному файлу нужно задать расширение •.ps, а сохраняемый файл получит расширение •.eps. Ключи -B -g инициируют вычисление BoundingBox, а ключ -f необходим, если сохраняемый файл уже существует.

Дистрибутивы TEX содержат программу pdfcrop, обрезающую пустые поля pdf-рисунков. Большое число ключей гибко настраивает ее действие, но в большинстве случаев достаточно запустить ее, указав только имя обрабатываемого файла:

```
pdfcrop <имя файла>.
```

Скорректированный рисунок запишется в новый файл, в имени которого появится «суффикс» -crop. Для работы этой программы в операционной системе должен быть установлен интерпретатор Perl.

A.7. Программа xindex

Программа xindex, представляющая собой набор скриптов, написанных на языке Lua, запускается из командной строки, например:

```
xindex -a -i -c RU -l RU •.idx
```

Имя файла «•» используется для записи указателя и описания выполненных действий в файлы •.ind и •.ilg. Работу программы настраивает ряд ключей, ни один из которых не является обязательным [82]:

-h, --help — печатает список ключей;

-V, --version — печатает номер версии программы;

-q, --quiet — сокращает сообщения, выдаваемые на консоль и записываемые в •.ilg файл;

-v, -vv, -vvv — задает уровень детализации сообщений в •.ilg файле;

-c, --config — задает имя файла загружаемой конфигурации (по умолчанию загружается файл xindex-cfg.lua);

-e, --escapechar — заменяет символ, защищающий символы `!@"|`, например, `-e >` (по умолчанию им служит двойная кавычка);

-a, --no_casesensitive — устанавливает сортировку без разделения заглавных и строчных букв;

-i, --ignoreSpace — устанавливает сортировку без учета пробелов;

-x,--no_UCA — отменяет использование алгоритма сортировки пакета Lua-UCA [96];

-g, --no_pagenumber — подавляет печать номеров страниц;

-n, --noheadings — подавляет печать заголовков рубрик;

-b, --no — отменяет присваивание меток рубрикам, по умолчанию получающим метку `\label{L-xindex-•}`;

-p, --prefix — заменяет префикс L в метках рубрик;

-l, --language — настраивает язык печати названий рубрик для символов и чисел;

-k, --checklang — автоматически настраивает язык, объявленный пакетом babel в •.aux файле;

-o, --output — задает имя файла для записи указателя;

-s, --use_stdin — встраивает xindex в конвейер, заменяя входной файл выводом другой программы;

-f, --fix_hyperref — устраняет ошибки оформления гиперссылок в элементах, содержащих символ «|».

Ключ -l настраивает заголовки рубрик для символов и чисел и действие не используемого в российской полиграфии параметра fCompress:

```
indexheader = { RU = {"Символы", "Числа", "russian"}, }
folium = { RU = { "" , "" }, }
```

Ключевое слово russian указывает основной язык указателя. Если пакет babel установил его в качестве основного, ключи -k или –checklang автоматически загрузят его настройки.

xindex-RU.lua

```
PageDelimiter   = ","
compressPages   = true
fCompress       = false
minCompress     = 2
rangeSymbol     = "--"
numericPage     = true
pageNoPrefixDel = ""
idxnewletter    = "\\textbf"
sublabels       = {"", "--- \\-", "--- --- \\-", "--- --- --- \\-"}
envStart        = "\\begin{theindex}"
indexOpening    = "\\makeatletter\z
                   \\def\\subitem{\\@idxitem}\z
                   \\def\\subsubitem{\\@idxitem}\z
                   \\def\\subsubsubitem{\\@idxitem}\z
                   \\makeatother"
envStop         = "\\end{theindex}"
```

Названия конфигурационных файлов имеют структуру xindex-•.lua, где часть • используется для адресации к файлу ключом -c. Рассмотрим настройки xindex на примере представленного выше конфигурационного файла русского языка xindex-RU.lua.

Настройки в виде `<<параметр = значение>>` перечисляются в начале файла, а в его конце указываются правила сортировки алфавитов латиницы (см. далее). Символ «\» является специальным в языке Lua, поэтому в конфигурационных файлах он дублируется.

Параметр PageDelimiter устанавливает символ, разделяющий список страниц в элементах указателя. Параметр compressPages, принимающий значения true или false, регулирует сокращение последовательности страниц до диапазонов. По умолчанию, если в указатель попали ссылки на страницы 3, 4, 5, 6, они будут сокращены до 3–6. Минимальное число страниц в последовательностях (без учета первой) задает параметр minCompress, а параметр rangeSymbol указывает символ, связывающий страницы диапазона. Параметр fCompress задает альтернативный способ обозначения диапазона страниц, принятый в некоторых странах. Параметр numericPage, принимающий значения true или false, позволяет включать в указатель страницы, заданные не арабскими цифрами, например II, IV, VI-17, а параметр pageNoPrefixDel устанавливает символ разделяющий номера страниц, заданных разным типом цифр.

Параметр idxnewletter задает шрифт печати заголовков рубрик. По умолчанию используется жирный шрифт, но его можно заменить любым другим:

`\newcommand\idxnewletter[1]{\textbf{\large\textit{#1}}}`,
idxnewletter = `"\\idxnewletter"`.

Данные настройки используют команду `\idxnewletter`, печатающую заголовки жирным крупным курсивом. Так как она отсутствует в арсенале LaTeX, ее нужно определить в преамбуле рукописи.

Параметры envStart и envStop задают окружение, используемое для печати указателя. В параметре sublabels через запятую перечисляются команды LaTeX, выполняемые перед печатью элементов определенного уровня. В нашем случае перед элементами первого уровня ничего не выводится, а перед элементами второго, третьего и четвертого уровня вставляются одно, два и три тире. Параметр indexOpening позволяет корректировать команды, печатающие элементы указателя. Их определения, заданные в файле xindex-RU.lua, убирают лишние отступы, по умолчанию вставляемые перед вложенными элементами. Команда `\z` в языке Lua сшивает строки, убирая пробелы.

xindex-description.lua

```
pageNoPrefixDel = ""
sublabels       = {"", "\\item", "\\item", "\\item"}
envStart        = "\\begin{description}"
indexOpening    = "\\makeatletter\z
                    \\def\\subitem{\\@idxitem}\z
                    \\def\\subsubitem{\\@idxitem}\z
                    \\def\\subsubsubitem{\\@idxitem}\z
                    \\makeatother\z
                    \\def\\indexspace{\\relax}"
envStop         = "\\end{description}"

alphabet_lower = { { ' ' }, { 'a', 'á', 'à', 'ä', 'â', 'å', 'æ', },
   { 'b' },{ 'c', 'ç' }, { 'd' }, { 'e', 'é', 'è', 'ë', 'ê' },
   { 'f' }, { 'g' },{ 'h' }, { 'i', 'í', 'ì', 'î', 'ï' }, { 'j' },
   { 'k' }, { 'l' }, { 'm' }, { 'n', 'ñ' },
   { 'o', 'ó', 'ò', 'ö', 'ô', 'ø', 'œ', 'ø'}, { 'p' }, { 'q' },
   { 'r' }, { 's', 'š' }, { 'ss', 'ß' }, { 't' },
   { 'u', 'ú', 'ù', 'ü' , 'û'}, { 'v' }, { 'w' },
   { 'x' }, { 'y', 'ý', 'ÿ' }, { 'z', 'ž' } }
alphabet_upper = { { ' ' }, { 'A', 'Á', 'À', 'Ä', 'Å', 'Æ', 'Â' },
   { 'B' }, { 'C', 'Ç' }, { 'D' }, { 'E', 'È', 'É', 'Ë', 'Ê' },
   { 'F' }, { 'G' }, { 'H' }, { 'I', 'Í', 'Ì', 'Ï', 'Î' }, { 'J' },
   { 'K' }, { 'L' }, { 'M' }, { 'N', 'Ñ' },
   { 'O', 'Ó', 'Ò', 'Ö', 'Ø','Œ', 'Ø', 'Ô'}, { 'P' }, { 'Q' },
   { 'R' }, { 'S', 'Š' }, { 'T' }, { 'U', 'Ú', 'Ù', 'Ü', 'Û' },
   { 'V' }, { 'W' }, { 'X' }, { 'Y', 'Ý', 'Ÿ' }, { 'Z', 'Ž' } }
```

Специальная конфигурация xindex-description.lua используется для верстки глоссариев. Заголовки рубрик и номера страниц в них не нужны, поэтому устанавливать язык не требуется. Значения семи первых параметров конфигурационного файла на верстку не влияют, поэтому их можно опустить. Параметр pageNoPrefixDel, не влияющий на результаты верстки, необходим для выполнения сортировки. Сортировка запускается с ключами

```
xindex -c description -l RU -a -n -g •.idx
```

Ключи -n и -g подавляют вывод заголовков рубрик и номеров страниц, ключ -a убирает различие строчных и заглавных букв, а ключ -c description загружает конфигурационный файл xindex-description.cfg. Его

параметры envStart и envStop помещают отсортированный список в окружение description, а параметр sublabels ставит перед элементами списка команду `\item`, превращающую их в записи данного окружения. В параметр indexOpening добавлена установка команды `\indexspace` в значение `\relax` — «ничего не делать». По умолчанию данная команда вставляет отбивку между разными рубриками указателя, ненужную в списке терминов.

В списке аббревиатур необходимо также убрать все отбивки окружения description, иначе он будет выглядеть рыхлым. Это делает конфигурационный файл xindex-abbreviations.lua, идентичный файлу xindex-description.lua во всем, кроме значения параметра indexOpening, в который добавлено зануление длин всех пробелов, кроме отбивки элементов окружения description левого края страницы:

xindex-abbreviations.lua

```
            ...
indexOpening = "\\makeatletter\z
                \\def\\subitem{\\@idxitem}\z
                \\def\\subsubitem{\\@idxitem}\z
                \\def\\subsubsubitem{\\@idxitem}\z
                \\makeatother\z
                \\parsep 0pt \\itemsep 0pt \\partopsep 0pt\z
                \\topsep 0pt \\parskip 0pt \\labelsep  0pt\z
                \\def\\indexspace{\\relax}"
            ...
```

Файлы xindex-description.lua и xindex-abbreviations.lua не поставляются вместе с программой xindex. Их нужно создать самостоятельно, скопировать в них представленный набор параметров и поместить в папку с компилируемым документом или папку с локальными ресурсами программы xindex (см. рис. А.2).

А.8. Программы xindy и texindy

Программу xindy верстающую указатели, запускает программа texindy которая передает ей данные, собранные компилятором LaTeX, месте с настройками, позволяющими разбирать специфический синтаксис описаний элементов указателей. Без этих настроек программа xindy с версткой указателя не справится, так как сама по себе она «не знакома» ни синтаксисом записей, ни синтаксисом LaTeX. Настрой-

ки представляют собой код, написанный на языке Lisp, разбитый на большое число файлов, имеющих расширение •.xdy.

Программа texindy адаптирует xindy для обработки команд LaTeX, вследствие чего команды изменения атрибутов шрифта не влияют на сортировку. Например, у термина `\index{\textit{example}}`, напечатанного курсивом, команда `\textit` игнорируется, и он будет сортироваться как `\index{example}`. В именах не игнорируемых команд отбрасываются символы «\», вдобавок у команд отбрасываются параметры, поэтому xindy занесет команду `\index{\command}` в рубрику указателя «C» и отсортирует вместе с ней команду `\index{\command [...]}`.

Работой texindy и xindy управляют следующие ключи:

-V, --version — печатает номер версии программы и основных модулей;

-?, -h, --help — печатает список ключей;

-q, --quiet — сокращает сообщения, выдаваемые на консоль и записываемые в •.ilg файл;

-v, --verbose — повышает уровень детализации сообщений;

-d, --debug — выводит информацию различных аспектов сортировки, задаваемых ключевыми словами: script — сообщения запускаемых скриптов; keep_tmpfiles — сохранение «промежуточных» файлов; markup — сообщения об обработке команд LaTeX; level=0–3 — уровень детализации сообщений (параметр -d можно использовать многократно, задавая ключевые слова по отдельности);

-r, --no-ranges — удаляет символ «–», разделяющий диапазоны страниц;

-C, --codepage — устанавливает кодировку сортируемого файла;

-L, --language — загружает настройки указанного языка;

-M, --module — загружает дополнительные настройки (параметр -M можно использовать многократно, чтобы комбинировать различные настройки, разбив их на несколько файлов);

-o, --out—file — задает имя файла для записи указателя;

-t, --log-file — задает имя файла для записи сообщений;

-i, --stdin — встраивает texindy в конвейер, заменяя входной файл выводом другой программы.

В простейшем случае при запуске texindy необходимо задать язык и кодировку записей. Приведем пример генерации русского указателя в кодировке utf8 из данных, находящихся в файле •.idx:

```
texindy.exe -C utf8 -L russian •.idx
```

Сверстанный указатель будет записан в файл с тем же именем • и расширением ind. При таком запуске правила сортировки русского текста добавляются к загружаемым по умолчанию правилам сортировки латиницы, поэтому русские и английские термины сортируются корректно. В то же время русская часть указателя будет разбита на рубрики, а английские термины будут выведены общим списком. Чтобы исправить эту ошибку, нужно составить и загрузить *общую* таблицу сортировки. Приведем необходимые настройки, которые можно скопировать в файл rueng.xdy и поместить в папку с компилируемым документом или локальными ресурсами xindy (см. рис. А.2):

rueng.xdy

```
;; Sorting order
(define-letter-groups ("A" "B" "C" "D" "E" "F" "G"
    "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S"
     "T" "U" "V" "W" "X" "Y" "Z"
     "А" "Б" "В" "Г" "Д" "Е" "Ж" "З" "И" "Й" "К" "Л"
     "М" "Н" "О" "П" "Р" "С" "Т" "У" "Ф" "Х" "Ц" "Ч"
     "Ш" "Щ" "Ъ" "Ы" "Ь" "Э" "Ю" "Я" ))

;; Common alphabet: ё = е
(define-rule-set "alphabetize"
     :rules (("а" "А") ("б" "Б") ("в" "В") ("г" "Г")
     ("д" "Д") ("е" "Е") ("Ё" "Е") ("ё" "Е") ("ж" "Ж")
     ("з" "З") ("и" "И") ("й" "Й") ("к" "К") ("л" "Л")
     ("м" "М") ("н" "Н") ("о" "О") ("п" "П") ("р" "Р")
     ("с" "С") ("т" "Т") ("у" "У") ("ф" "Ф") ("х" "Х")
     ("ц" "Ц") ("ч" "Ч") ("ш" "Ш") ("щ" "Щ") ("ъ" "Ъ")
     ("ы" "Ы") ("ь" "Ь") ("э" "Э") ("ю" "Ю") ("я" "Я")
     ("a" "A") ("b" "B") ("c" "C") ("d" "D") ("e" "E")
     ("f" "F") ("g" "G") ("h" "H") ("i" "I") ("j" "J")
     ("k" "K") ("l" "L") ("m" "M") ("n" "N") ("o" "O")
     ("p" "P") ("q" "Q") ("r" "R") ("s" "S") ("t" "T")
     ("u" "U") ("v" "V") ("w" "W") ("x" "X") ("y" "Y")
     ("z" "Z")))
(define-rule-set "reorderyo"
   :rules (("Е" "8") ("е" "8") ("Ё" "9") ("ё" "9")))

;; Handling special characters
(define-rule-set "ignore-special"
   :rules  (("." "{}" :string) ("?" "{}" :string)
```

```
 ("!" "{}" :string) ("{}'{}" "{}" :string) ("{}-{}" "{}" :string)
     ("{" "{}" :string) ("}" "{}" :string)))

;; Sorting cycles
(use-rule-set :run 0
   :rule-set ("alphabetize" "ignore-special"))
(use-rule-set :run 1
   :rule-set ("reorderyo" "ignore-special"))

;; Underlining page nubers
(define-attributes (( "underline" )) )
(markup-locref  :open "\underline{" :close "}"
   :attr  "underline")
```

В данные настройки добавлена возможность подчеркивать страницы в элементах указателей с помощью команд `\index{...|underline}`. Файл с дополнительными настройками загружает ключ -M, за которым должно следовать его имя, указываемое без расширения:

```
texindy -C utf8 -L russian -M rueng •.idx.
```

Пакет hyperref позволяет преобразовать ссылки указателя в гиперссылки. Для этого он вводит команду `\hyperpage{`*номер страницы*`}`, аргументом которой является номер страницы. Настройки, встраивающие ее в ресурсы xindy, можно собрать в файл hyperref.xdy, который следует загружать командой

```
texindy -C utf8 -L russian -M rueng -M hyperref •.idx
```

hyperref.xdy

```
;; Make hyperrefs to page numbers
(define-attributes (( "underline" )) )
(markup-locref  :open "\underline{\hyperpage{" :close "}}"
   :attr  "underline")
(markup-locref  :open "\textbf{\hyperpage{" :close "}}"
  :attr  "textbf")\\
(markup-locref  :open "\textit{\hyperpage{" :close "}}"
  :attr  "textit")
(markup-locref  :open "\hyperpage{" :close "}"
   :attr  "default")
```

Еще раз подчеркнем, что приведенные настройки отсутствуют в стандартных ресурсах программы xindy. Скопируйте их в файлы и по-

местите их в папку .../texmf-local/xindy, в которой xindy ищет локальные ресурсы (см. рис. A.2).

A.9. Скрипт splitindex

Скрипт splitindex используется пакетом imakeidx для разделения общего файла с данными указателей на файлы с данными отдельных указателей. Он может не только разделять данные, но и передавать их сортирующим программам, однако пакет imakeidx настраивает сортировку сам. По умолчанию imakeidx запускает скрипт с ключом, отменяющим вызов других программ:

```
splitindex -m "" ●.idx
```

В то же время, не ограничивая действий splitindex, он позволяет настроить его работу, передав ему нужный список ключей с помощью команды `\splitindexoptions{`*список ключей*`}`, которую следует поместить в преамбуле. Ключи, описанные в руководстве пакета splitindex [97], задают регулярные выражения, служащие для разделения и сохранения данных.

К сожалению, splitindex содержится в дистрибутивах TEX только в виде Perl-скрипта, поэтому его работа в ОС Windows осложнена отсутствием интерпретатора Perl. Учитывая это, автор программы splitindex разработал также код на языках C и Java. Вы можете найти его в TEX-архиве по адресу:

`https://www.ctan.org/tex-archive/macros/latex/contrib/splitindex`.

Компилировать программы придется самостоятельно.

Приложение Б

Символы и буквы текста

Б.1. Восьмибитные кодировки

Вследствие малой разрядности процессоров для печати текста раньше использовались различные 8-битные кодировки, вмещавшие всего 256 символов, из которых первые 128 (ASCII символы) являются общими для всех кодировок, а остальные 128 предназначены для букв национальных алфавитов. Среди кодировок, использовавшихся наиболее часто, выделим cp1252 (Windows) и iso8859-1 (международная), а также cp1251 (Windows). Две первые включают буквы европейских языков на основе латиницы, а последняя, обеспечивающая поддержку русского языка, до сих пор используется в русифицированных версиях Windows наряду с кодировкой unicode. Современные операционные системы используют кодировку unicode, имеющую три полноценные модификации UTF8, UTF16 и UTF32, различающиеся формой представления кодов символов.

В разд. 2.6.4 уже говорилось, что компилятор pdflatex оперирует несколькими кодировками. В начале верстки, в соответствии с кодировкой utf8 и дополнительными кодировками, загружаемыми различными пакетами, символы переводятся в команды. Во время верстки, чтобы получить информацию о параметрах символов, используется набор привязанных к командам восьмибитных кодировок. В сверстанном документе, в соответствии с кодировками печатных шрифтов, команды

заменяются глифами. Работу с восьмибитными кодировками обеспечивают файлы с именами XXenc.def и XXenc.dfu, где XX — наименование кодировки. Файлы с расширением •.def содержат привязку команд к 8-битным кодировкам, а файлы с расширением •.dfu — к кодировке unicode.

Восьмибитные кодировки загружает пакет fontenc [98]:

\usepackage[список кодировок]{fontenc}.

Кодировки перечисляются списком через запятую в параметре команды \usepackage. Последняя из них становится активной. Ее символы используются «непосредственно», а символы других кодировок, чтобы не печатать их командами, нужно дополнительно объявить декларациями:

\DeclareTextSymbolDefault{команда}{кодировка} .

После объявления основной буквы можно использовать все ее акцентированные варианты.

Поясним сказанное примером. Чтобы составить полную таблицу кириллицы, для верстки данной книги использовалась загрузка:

\usepackage[OT2,X2,T2B,T2C,T2A]{fontenc},

а затем в преамбуле объявлялись буквы, отсутствующие в кодировке T2A. Например, фита была объявлена командой

\DeclareTextSymbolDefault{\CYRFITA}{OT2}.

Символ любой загруженной кодировки без объявления можно напечатать командой

\UseTextSymbol{кодировка}{команда} .

Напомним, что декларации

\fontencoding{кодировка}\selectfont

позволяют активировать любую загруженную кодировку.

Любые символы активной кодировки, в том числе специальные, печатает команда

\symbol{номер}

Ее аргументом является номер символа, задаваемый десятеричным числом от нуля до 255.

Б.2. Кодировки T1, T2A и TS1

Кодировки T1 и T2A, используемые наиболее часто, представлены в табл. Б.1.

Таблица Б.1

Таблица кодировок T1 и T2A

ASCII: 20–7E

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	`	´	ˆ	˜	¨	˝	˚	ˇ	˘	¯	·	¸	˛	,	I	‹	›
1	“	”	„	^	«	`	»	¨	–	—		ı	ȷ	ff	fi	fl	ffi	ffl
2	␣	!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	-

T1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ă	Ą	Ć	Č	Ď	Ě	Ę	Ğ	Ĺ	Ľ	Ł	Ń	Ň	Ŋ	Ő	Ŕ
9	Ř	Ś	Š	Ş	Ť	Ţ	Ű	Ů	Ÿ	Ź	Ž	Ż	IJ	İ	đ	§
A	ă	ą	ć	č	ď	ě	ę	ğ	ĺ	ľ	ł	ń	ň	ŋ	ő	ŕ
B	ř	ś	š	ş	ť	ţ	ű	ů	ÿ	ź	ž	ż	ij	¡	¿	£
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Đ	Ñ	Ò	Ó	Ô	Õ	Ö	Œ	Ø	Ù	Ú	Û	Ü	Ý	Þ	SS
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	œ	ø	ù	ú	û	ü	ý	þ	ß

T2A

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	Ґ	Ғ	Ҕ	Ҳ	Һ	Ӂ	Ӌ	Љ	Ї	Қ	Ҡ	Ҟ	Æ	Ң	Ҥ	S
9	Ө	Ҫ	Ў	Ү	Ұ	Ҳ	Ц	Ч	Ҷ	Є	Ә	Њ	Ё	№	¤	§
A	ґ	ғ	ҕ	ҳ	h	ӂ	ӌ	љ	ї	қ	ҡ	ҟ	æ	ң	ҥ	s
B	ө	ҫ	ў	ү	ұ	ҳ	ц	ч	ҷ	є	ә	њ	ё	„	«	»
C	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
D	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
E	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

Символы, занимающие первые сто двадцать восемь позиций с шестнадцатеричными кодами 00 – 7F, являются общими для обеих кодировок, кроме тех, что выделены фоном. Среди них левые входят в кодировку T1, а правые — в T2A.

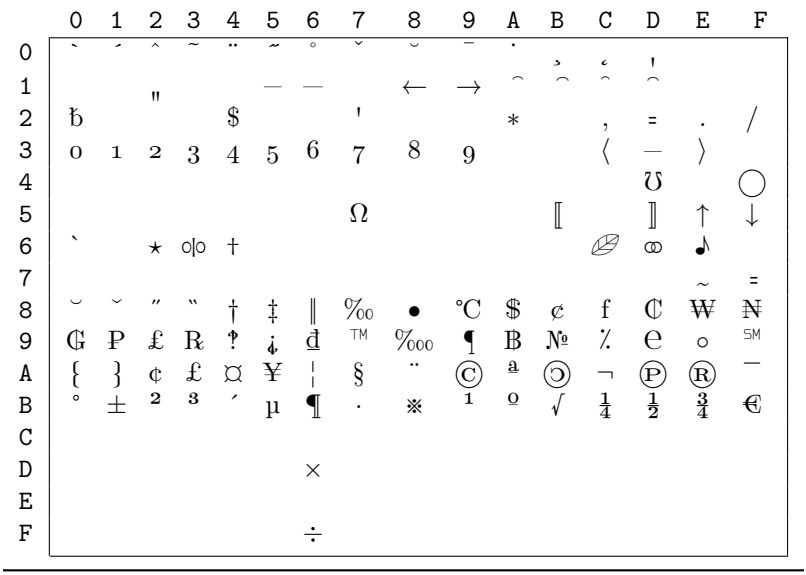
ASCII символы с кодами 20 – 7E содержат латинский алфавит и набор символов, необходимых для поддержки синтаксиса команд LaTeX.

В средней и нижней части таблицы приведены символы с кодами 80 – FF, различающиеся в кодировках T1 и T2A.

В табл. Б.2 показаны символы, используемые в наборе текста на разных языках, собранные в кодировку TS1. Она содержит цифры старого стиля, выводимые командой `\oldstylenums{•}`.

Таблица Б.2

Таблица кодировки TS1



Б.3. Символы текста

В табл. Б.3 собраны небуквенные символы кодировок TS1, T1 и T2•. В колонках, маркированных значком •, находятся символы, в колонках с заголовком U+# приведена их привязка к кодировке unicode, а следующие колонки содержат команды, печатающие символы.

При использовании компиляторов LaLaTeX и XeLaTeX набор символов, вводимых командами, сокращается до выделенных фоном.

Таблица Б.3

Символы текста

•	U+#	Команды	•	U+#	Команды
	200C	\textcompwordmark		A0	\nobreakspace
␣	2423	\textvisiblespace	°	B0	\textdegree
–	2014	--, \textendash	—	2013	---, \textemdash
†	2020	\textdagger, \dag	‡	2021	\textdaggerdbl, \ddag
†	271D	\textdied	*	2605	\textborn
%	25	\%	#	23	\#
&	26	\&	_	5F	_, \textunderscore
$		\textdollaroldstyle	$	24	\$, \textdollar
/	2044	\textfractionsolidus	\	5C	\textbackslash
{	7B	\{, \textbraceleft	}	7D	\}, \textbraceright
{	2045	\textlquill	}	2046	\textrquill
⟦	301A	\textlbrackdbl	⟧	301B	\textrbrackdbl
⟨	2329	\textlangle	⟩	232A	\textrangle
<	3C	<, \textless	>	3E	>, \textgreater
‹	2039	\guilsinglleft	›	203A	\guilsinglright
«	AB	<<, \guillemotleft, \guillemetleft			
»	BB	>>, \guillemotright, \guillemetright			
‧	2E40	\textdblhyphen, \textdblhyphenchar			
←	2190	\textleftarrow	→	2192	\textrightarrow
↑	2191	\textuparrow	↓	2193	\textdownarrow
|	7C	\textbar	¦	A6	\textbrokenbar
‖	2016	\textbardbl	¬	ac	\textlnot
−	2212	\textminus	×	D7	\texttimes
÷	F7	\textdiv	±	B1	\textpm
※	203B	\textreferencemark	-	2010	-
*	204E	\textasteriskcentered	№	2116	\textnumero
·	B7	\textperiodcentered	…	2026	\textellipsis
◦	25E6	\textopenbullet	•	2022	\textbullet
¿	BF	\textquestiondown	¡	A1	\textexclamdown
‽	203D	\textinterrobang	¡	2E18	\textinterrobangdown
¶	B6	\textparagraph	¶		\textpilcrow
§	A7	\textsection	♪	266A	\textmusicalnote
¤	A4	\textcurrency	⁒	2052	\textdiscount
‰	2030	\textperthousand	‱	2031	\textpertenthousand

Окончание табл. Б.3

•	U+#	Команды	•	U+#	Команды
฿	E3F	\textbaht	₵	20A1	\textcolonmonetary
¢	A2	\textcent	¢		\textcentoldstyle
₫	20AB	\textdong	€	20AC	\texteuro
ƒ	192	\textflorin	₲	20B2	\textguarani
£	A3	\textsterling	£	20A4	\textlira
₦	20A6	\textnaira	₱	20B1	\textpeso
₩	20A9	\textwon	¥	A5	\textyen
µ	B5	\textmu	♭	2422	\textblank
Ω	2126	\textohm	℮	212E	\textestimated
℧	2127	\textmho	℞	211E	\textrecipe
√	221A	\textsurd	℃	2103	\textcelsius
⚭	26AD	\textmarried	⚮	26AE	\textdivorced
0		\textzerooldstyle	1		\textoneoldstyle
2		\texttwooldstyle	3		\textthreeoldstyle
4		\textfouroldstyle	5		\textfiveoldstyle
6		\textsixoldstyle	7		\textsevenoldstyle
8		\texteightoldstyle	9		\textnineoldstyle
ª	AA	\textordfeminine	º	BA	\textordmasculine
¹	B9	\textonesuperior	½	BD	\textonehalf
²	B2	\texttwosuperior	¼	BC	\textonequarter
³	B3	\textthreesuperior	¾	BE	\textthreequarters
SM	2120	\textservicemark	TM	2122	\texttrademark
©	A9	\textcopyright	ⓒ	1F12F	\textcopyleft
Ⓟ	2117	\textcircledP	®	AE	\textregistered
◯	25EF	\textbigcircle	⦿		\textcircled{•}
¯	AF	\textasciimacron	🍃	1F653	\textleaf
’	27	\textquotesingle	‚	201A	\quotesinglbase
”	22	\textquotedbl	„	201E	\quotedblbase
‘	2018	\textquoteleft	’	2019	\textquoteright
“	201C	\textquotedblleft	”	201D	\textquotedblright
¨	A8	\textasciidieresis	″	2DD	\textacutedbl
´	B4	\textasciiacute	˘	2D8	\textasciibreve
ˆ	2C6	\textasciicircum	ˇ	2C7	\textasciicaron
‘	60	\textasciigrave	̏	2F5	\textgravedbl
˜	2F7	\texttildelow	˜	2DC	\textasciitilde

Б.4. Диакритические знаки

Большое количество букв латиницы и кириллицы имеют диакритические знаки, расставляемые с помощью команд, собранных в табл. Б.4. В колонках, помеченных символом •, приведены примеры акцентированных букв.

Таблица Б.4

Команды расстановки диакритических знаков

•	Команды	•	Команды	•	Команды	•	Команды
ò	`\`o`	ó	`\'o`	ȯ	`\.o`	ö	`\"o`
ő	`\H{o}`	o͒	`\C{o}`	ô	`\^o`	ǒ	`\v{o}`
o͚	`\r{o}`	ŏ	`\U{o}`	ŏ	`\u{o}`	o͡	`\f{o}`
ō	`\=o`	õ	`\~o`	o͡o	`\t{oo}`	o̲	`\b{o}`
o̧	`\c{o}`	ọ	`\d{o}`	o̧	`\k{o}`	ọ	`\textcommabelow{o}`

Б.5. Латиница

Буквы латиницы представлены в табл. Б.5. При работе с pdflatex буквы, выделенные фоном, вводятся командами, а остальные — в виде символов. Буквы с точкой снизу отсутствуют в кодировках utf8 и T1, другие выделенные буквы недоступны в кодировке T2A.

Таблица Б.5

Буквы алфавитов латиницы

•	U+#	Команды	•	U+#	Команды	•	U+#	Команды	•	U+#	Команды
À	C0	`\`A`	à	E0	`\`a`	Á	C1	`\'A`	á	E1	`\'A`
Ä	C4	`\"A`	ä	E4	`\"A`	Å	C5	`\r{A}`	å	E5	`\r{a}`
Ã	C3	`\~A`	ã	E3	`\~a`	Ā	100	`\=A`	ā	101	`\=a`
Â	C2	`\^A`	â	E2	`\^a`	Ǎ	1CD	`\v{A}`	ǎ	1CE	`\u{a}`
Ă	102	`\u{A}`	ă	103	`\u{a}`	Ą	104	`\k{A}`	ą	105	`\k{a}`
Ạ	1EA0	`\d{A}`	ạ	1EA1	`\d{a}`	Æ	C6	`\AE`	æ	E6	`\ae`
Ǣ	1E2	`\=\AE`	ǣ	1E3	`\=\ae`	Ḃ	1E02	`\.B`	ḃ	1E03	`\.b`
Ḅ	1E04	`\d{B}`	ḅ	1E05	`\d{b}`	Ć	106	`\'C`	ć	107	`\'c`
Ĉ	108	`\^C`	ĉ	109	`\^c`	Č	10C	`\v{C}`	č	10D	`\v{c}`

Продолжение табл. Б.5

•	U+#	Команды	•	U+#	Команды	•	U+#	Команды	•	U+#	Команды
Ċ	10A	\.C	ċ	10B	\.c	Ç	C7	\c{C}	ç	E7	\c{c}
Ď	10E	\v{D}	ď	10F	\v{d}	Ḍ	1E0C	\d{D}	ḍ	1E0D	\d{d}
Đ	D0	\DH	ð	F0	\dh	Đ	110	\DJ	đ	111	\dj
È	C8	\`E	è	E8	\`e	É	C9	\'E	é	E9	\'e
Ė	116	\.E	ė	117	\.e	Ë	CB	\"E	ë	EB	\"e
Ē	112	\=E	ē	113	\=e	Ĕ	114	\u{E}	ĕ	115	\u{e}
Ê	CA	\^E	ê	EA	\^e	Ě	11A	\v{E}	ě	11B	\v{e}
Ę	118	\k{E}	ę	119	\k{e}	Ẹ	1EB8	\d{E}	ẹ	1EB9	\d{e}
Ǵ	1F4	\'G	ǵ	1F5	\'g	Ġ	120	\.G	ġ	121	\.g
Ḡ	1E20	\=E	ḡ	1E21	\=g	Ğ	11E	\u{G}	ğ	11F	\u{g}
Ĝ	11C	\^G	ĝ	11D	\^g	Ǧ	1E6	\v{G}	ǧ	1E7	\v{g}
Ģ	122	\c{G}	g	123	\c{g}	Ģ	---	\d{G}	g	---	\d{g}
Ĥ	124	\^H	ĥ	125	\^h	Ḥ	1E24	\d{H}	ḥ	1E25	\d{h}
Ħ	126	\Hwithstroke				ħ	127	\hwithstroke			
Ì	CC	\`I	ì	EC	\`i	Í	CD	\'I	í	ED	\'i
İ	130	\.I	ı	131	\i	Ï	CF	\"I	ï	EF	\"i
Ĩ	128	\~I	ĩ	129	\~i	Ī	12A	\=I	ī	12B	\=i
Î	CE	\^I	î	EE	\^i	Ǐ	1CF	\v{I}	ǐ	1D0	\v{i}
Ĭ	12C	\u{I}	ĭ	12D	\u{i}	Į	12E	\k{I}	į	12F	\k{i}
Ị	1ECA	\d{I}	ị	1ECB	\d{i}	IJ	132	\IJ	ij	133	\ij
Ĵ	134	\^J	ĵ	135	\^j	ǰ	1F0	\v{j}	ȷ	237	\j
Ǩ	1E8	\v{K}	ǩ	1E9	\v{k}	Ķ	136	\c{K}	ķ	137	\c{k}
Ḳ	1E32	\d{K}	ḳ	1E33	\d{k}	Ĺ	139	\'L	Í	13A	\'l
Ľ	13D	\v{L}	ľ	13E	\v{l}	Ł	141	\L	ł	142	\l
Ļ	13B	\c{L}	ļ	13C	\c{l}	Ḷ	1E36	\d{L}	ḷ	1E37	\d{l}
Ṃ	1E42	\d{M}	ṃ	1E43	\d{m}	Ń	143	\'N	ń	144	\'n
Ñ	D1	\~N	ñ	F1	\~n	Ň	147	\v{N}	ň	148	\v{n}
Ņ	145	\c{N}	ņ	146	\c{n}	Ṇ	1E46	\d{N}	ṇ	1E47	\d{n}
Ŋ	14A	\NG	ŋ	14B	\ng	Ø	D8	\O	ø	F8	\o
Ò	D2	\`O	ò	F2	\`o	Ó	D3	\'O	ó	F3	\'o
Ô	D4	\^O	ô	F4	\^i	Ǒ	1D1	\v{O}	ǒ	1D2	\v{o}
Ŏ	14E	\u{O}	ŏ	14F	\u{o}	Ō	14C	\=O	ō	14D	\=o
Ö	D6	\"O	ö	F6	\"o	Ő	150	\H{O}	ő	151	\H{o}

Окончание табл. Б.5

•	U+#	Команды	•	U+#	Команды	•	U+#	Команды	•	U+#	Команды
Õ	D5	\~O	õ	F5	\~o	Ọ	1EA	\k{O}	ọ	1EB	\k{o}
Ọ	1ECC	\d{O}	ọ	1ECD	\d{o}	Œ	152	\OE	œ	153	\oe
Ŕ	154	\'R	ŕ	155	\'r	Ř	158	\v{R}	ř	159	\v{r}
Ŗ	156	\c{R}	ŗ	157	\c{r}	Ṛ	1E5A	\d{R}	ṛ	1E5B	\d{r}
Ś	15A	\'S	ś	15B	\'s	ß	1EDE	\SS	ß	DF	\ss
Ŝ	15C	\^S	ŝ	15D	\^s	Š	160	\v{S}	š	161	\v{s}
Ş	15E	\c{S}	ş	15F	\c{s}	Ṣ	1E62	\d{S}	ṣ	1E63	\d{s}
Ș	218	\textcommabelow{S}				ș	219	\textcommabelow{s}			
Ț	21A	\textcommabelow{T}				ț	21B	\textcommabelow{t}			
Ť	164	\v{T}	ť	165	\v{t}	Ţ	162	\c{T}	ţ	163	\c{t}
Ṭ	1E6C	\d{T}	ṭ	1E6D	\d{t}	Þ	DE	\TH	þ	FE	\th
Ù	D9	\`U	ù	F9	\`u	Ú	DA	\'U	ú	FA	\'u
Û	DB	\^U	û	FB	\^u	Ŭ	1D3	\v{U}	ŭ	1D4	\v{u}
Ŭ	16C	\u{U}	ŭ	16D	\u{u}	Ů	16E	\r{U}	ů	16F	\r{u}
Ü	DC	\"U	ü	FC	\"u	Ű	170	\H{U}	ű	171	\H{u}
Ũ	168	\~U	ũ	169	\~u	Ū	16A	\=U	ū	16B	\=u
Ų	172	\k{U}	ų	173	\k{u}	Ụ	1EE4	\d{U}	ụ	1EE5	\d{u}
Ṿ	1E7E	\d{V}	ṿ	1E7F	\d{v}	Ẉ	1E88	\d{W}	ẉ	1E89	\d{w}
Ŵ	174	\^W	ŵ	175	\^w	Ŷ	176	\^Y	ŷ	177	\^y
Ý	DD	\'Y	ý	FD	\'y	Ÿ	178	\"Y	ÿ	FF	\"y
Ȳ	232	\=Y	ȳ	233	\=y	Ỵ	1EF4	\d{Y}	ỵ	1EF5	\d{y}
Ź	179	\'Z	ź	17A	\'z	Ž	17D	\v{Z}	ž	17E	\v{z}
Ż	17B	\.Z	ż	17C	\.z	Ẓ	1E92	\d{Z}	ẓ	1E93	\d{z}

Б.6. Кириллица

Набор букв кириллицы очень велик, поэтому его перекрывают несколько кодировок: T2B, T2C, OT2 X2. В первой половине кодировок T2A, T2B и T2C находятся символы, общие для T1 и T2• кодировок, а во второй половине — русские буквы и различные наборы других букв кириллицы (см. для примера табл. Б.1). Чтобы охватить максимальное количество букв, определена кодировка X2, в которой кириллица занимает обе половины таблицы. В кодировке OT2, содержащей всего 128 символов, в настоящее время не используемой, кириллицей заменены латинские буквы и часть ASCII символов.

В табл. Б.6 собраны буквы кириллицы, большинство из которых поддерживается несколькими кодировками, указанными в последней колонке с помощью следующих сокращений: A – T2A, B – T2B, C – T2C, O – OT2, X – X2. Таким образом, аббревиатура ABOX означает, что буквы входят в кодировки T2A, T2B, OT2 и X2. В колонках с заголовком U+# представлены unicode-коды букв, а сами буквы и печатающие их команды — в соседних колонках.

Таблица Б.6

Буквы алфавитов кириллицы

•	U+#	Команды	•	U+#	Команды	Кодировки
А	410	\CYRA	а	430	\cyra	ABCOX
Ӑ	4D0	\U\CYRA	ӑ	4D1	\U\cyra	ABCOX
Ӓ	4D2	\"\CYRA	ӓ	4D3	\"\cyra	ABCOX
Æ	4D4	\CYRAE	æ	4D5	\cyrae	AX
Б	411	\CYRB	б	431	\cyrb	ABCOX
В	412	\CYRV	в	432	\cyrv	ABCOX
Г	413	\CYRG	г	433	\cyrg	ABCOX
Ѓ	403	\'\CYRG	ѓ	453	\'\cyrg	ABCOX
Ґ	490	\CYRGUP	ґ	491	\cyrgup	AX
Ғ	492	\CYRGHCRS	ғ	493	\cyrghcrs	ABX
Ҕ	494	\CYRGHC	ҕ	495	\cyrghc	BCX
Ґ	4F6	\CYRGDSC	ґ	4F7	\cyrgdsc	BX
Ғ̣	4FA	\CYRGDSCHCRS	ғ̣	4FB	\cyrgdschcrs	B
Д	414	\CYRD	д	434	\cyrd	ABCOX
Е	415	\CYRE	е	435	\cyre	ABCOX
È	400	\`\CYRE	è	450	\`\cyre	ABCOX
Ӗ	4D6	\U\CYRE	ӗ	4D7	\U\cyre	ABCOX
Ё	401	\CYRYO	ё	451	\cyryo	ABCOX
Ж	416	\CYRZH	ж	436	\cyrzh	ABCOX
Ӝ	4DC	\"\CYRZH	ӝ	4DD	\"\cyrzh	ABCOX
Ӂ	4C1	\U\CYRZH	ӂ	4C2	\U\cyrzh	ABCOX
Җ	496	\CYRZHDSC	җ	497	\cyrzhdsc	ABX
З	417	\CYRZ	з	437	\cyrz	ABCOX
Ӟ	4DE	\"\CYRZ	ӟ	4DF	\"\cyrz	ABCOX
Ҙ	498	\CYRZDSC	ҙ	499	\cyrzdsc	AX
И	418	\CYRI	и	438	\cyri	ABCOX
Ѝ	40D	\`\CYRI	ѝ	45D	\`\cyri	ABCOX
Ӣ	4E2	\=\CYRI	ӣ	4E3	\=\cyri	ABCOX

Продолжение 1 табл. Б.6

•	U+#	Команды	•	U+#	Команды	Кодировки
Й	4E4	\"\CYRI	й	4E5	\"\cyri	ABCOX
Й	419	\CYRISHRT	й	439	\cyrishrt	ABCOX
К	41A	\CYRK	к	43A	\cyrk	ABCOX
Ќ	40C	\'\CYRK	ќ	45C	\'\cyrk	ABCOX
Қ	49A	\CYRKDSC	қ	49B	\cyrkdsc	ABCX
Ҝ	49C	\CYRKVCRS	ҝ	49D	\cyrkvcrs	AX
Ҟ	49E	\CYRKHCRS	ҟ	49F	\cyrkhcrs	AX
Ҡ	4A0	\CYRKBEAK	ҡ	4A1	\cyrkbeak	AX
Ҷ	4C3	\CYRKHK	ҷ	4C4	\cyrkhk	BX
Л	41B	\CYRL	л	43B	\cyrl	ABCOX
Ԯ	4C5	\CYRLDSC	ԯ	4C6	\cyrldsc	BCX
Љ	409	\CYRLJE	љ	459	\cyrlje	ABOX
М	41C	\CYRM	м	43C	\cyrm	ABCOX
Ӎ	4CD	\CYRMDSC	ӎ	4CE	\cyrmdsc	CX
Н	41D	\CYRN	н	43D	\cyrn	ABCOX
Ң	4A2	\CYRNDSC	ң	4A3	\cyrndsc	ABCX
Ҥ	4A4	\CYRNG	ҥ	4A5	\cyrng	ABX
Њ	40A	\CYRNJE	њ	45A	\cyrnje	ABOX
Ӈ	4C7	\CYRNHK	ӈ	4C8	\cyrnhk	BCX
О	41E	\CYRO	о	43E	\cyro	ABCOX
Ӧ	4E6	\"\CYRO	ӧ	4E7	\"\cyro	ABCOX
Ө	4E8	\CYROTLD	ө	4E9	\cyrotld	ABCX
П	41F	\CYRP	п	43F	\cyrp	ABCOX
Ҧ	4A6	\CYRPHK	ҧ	4A7	\cyrphk	CX
Р	420	\CYRR	р	440	\cyrr	ABCOX
Ҏ	48E	\CYRRTICK	ҏ	48F	\cyrrtick	C
С	421	\CYRS	с	441	\cyrs	ABCOX
Ҫ	4AA	\CYRSDSC	ҫ	4AB	\cyrsdsc	AX
Т	422	\CYRT	т	442	\cyrt	ABCOX
Ҭ	4AC	\CYRTDSC	ҭ	4AD	\cyrtdsc	CX
Ҵ	4B4	\CYRTETSE	ҵ	4B5	\cyrtetse	CX
У	423	\CYRU	у	443	\cyru	ABCOX
Ӯ	4EE	\=\CYRU	ӯ	4EF	\=\cyru	ABCOX
Ӱ	4F0	\"\CYRU	ӱ	4F1	\"\cyru	ABCOX
Ӳ	4F2	\H\CYRU	ӳ	4F3	\H\cyru	ABCOX
Ў	40E	\CYRUSRT	ў	45E	\cyrushrt	ABX

Продолжение 2 табл. Б.6

•	U+#	Команды	•	U+#	Команды	Кодировки
Ф	424	\CYRF	ф	444	\cyrf	ABCOX
Х	425	\CYRH	х	445	\cyrh	ABCOX
Ҳ	4B2	\CYRHDSC	ҳ	4B3	\cyrhdsc	ABCX
Ӽ	4FC	\CYRHHC	ӽ	4FD	\cyrhhc	BX
Ӿ	4FE	\CYRHHCRS	ӿ	4FF	\cyrhhcrs	B
Ц	426	\CYRC	ц	446	\cyrc	ABCOX
Ч	427	\CYRCH	ч	447	\cyrch	ABCOX
Ӵ	4F4	\"\CYRCH	ӵ	4F5	\"\cyrch	ABCOX
Ҷ	4B6	\CYRCHRDSC	ҷ	4B7	\cyrchrdsc	ABCX
Ӌ	4CB	\CYRCHLDSC	ӌ	4CC	\cyrchldsc	BX
Ҹ	4B8	\CYRCHVCRS	ҹ	4B9	\cyrchvcrs	AX
Ш	428	\CYRSH	ш	448	\cyrsh	ABCOX
Щ	429	\CYRSHCH	щ	449	\cyrshch	ABCOX
Ъ	42A	\CYRHRDSN	ъ	44A	\cyrhrdsn	ABCOX
Ы	42B	\CYRERY	ы	44B	\cyrery	ABCOX
Ӹ	4F4	\"\CYRERY	ӹ	4F5	\"\cyrERY	ABCOX
Ь	42C	\CYRSFTSN	ь	44C	\cyrsftsn	ABCOX
Ҍ	48C	\CYRSEMISFTSN	ҍ	48D	\cyrsemisftsn	C
Э	42D	\CYREREV	э	44D	\cyrerev	ABCOX
Ӭ	4EC	\"\CYREREV	ӭ	4ED	\"\cyrerev	ABCOX
Є	404	\CYRIE	є	454	\cyrie	AOX
Ю	42E	\CYRYU	ю	44E	\cyryu	ABCOX
Я	42F	\CYRYA	я	44F	\cyrya	ABCOX
І	406	\CYRII	і	456	\cyrii	ABCOX
Ї	407	\CYRYI	ї	457	\cyryi	AX
Ј	408	\CYRJE	ј	458	\cyrje	ABCOX
Ѕ	405	\CYRDZE	ѕ	455	\cyrdze	ABCOX
Ү	4AE	\CYRY	ү	4AF	\cyry	ABX
Ұ	4B0	\CYRYHCRS	ұ	4B1	\cyryhcrs	AX
Һ	4BA	\CYRSHHA	һ	4BB	\cyrshha	ABCX
Ђ	402	\CYRDJE	ђ	452	\cyrdje	AOX
Ћ	40B	\CYRTSHE	ћ	45B	\cyrtshe	AOX
Џ	40F	\CYRDZHE	џ	45F	\cyrdzhe	ACOX
Ә	4D8	\CYRSCHWA	ә	4D9	\cyrschwa	ABCX
Ӛ	4DA	\"\CYRSCHWA	ӛ	4DB	\"\cyrschwa	ABCX
Ҽ	4BC	\CYRABHCH	ҽ	4BD	\cyrabhch	CX

Окончание табл. Б.6

•	U+#	Команды	•	U+#	Команды	Кодировки
Ҿ	4BE	\CYRABHCHDSC	ҿ	4BF	\cyrabhchdsc	CX
Ҩ	4A8	\CYRABHHA	ҩ	4A9	\cyrabhha	CX
Ӡ	4E0	\CYRABHDZE	ӡ	4E1	\cyrabhdze	BCX
Ѣ	462	\CYRYAT	ѣ	463	\cyryat	OX
Ѫ	46A	\CYRBYUS	ѫ	46B	\cyrbyus	X
Ѳ	472	\CYRFITA	ѳ	473	\cyrfita	O
Ѵ	474	\CYRIZH	ѵ	475	\cyrizh	OX
Ѷ	474	\C\CYRIZH	ѷ	475	\C\cyrizh	OX
			I	4C0	\CYRpalochka	ABCX

Приложение B

Математические символы

В формулах и таблицах гл. 4 использовались шрифты NewComputerModernMath, в данном разделе для сравнения приведены символы и алфавиты шрифтов STIX Two Math и рукописный шрифт XITS Math.

B.1. Математические акценты

В табл. B.1 представлены команды расстановки математических акцентов. В колонках, помеченных значком •, показаны примеры акцентированных букв.

Таблица B.1

Математические акценты

•	Команды	•	Команды	•	Команды
$\tilde{a}$	`\tilde{a}`	$\dot{a}$	`\dot{a}`	$\ddddot{a}$	`\dddot{a}`
$\bar{a}$	`\bar{a}`	$\ddot{a}$	`\ddot{a}`	$\dddddot{a}$	`\ddddot{a}`
$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\widehat{abc}$	`\widehat{abc}`
$\grave{a}$	`\grave{a}`	$\acute{a}$	`\acute{a}`	$\widetilde{abc}$	`\widetilde{abc}`
$\breve{a}$	`\breve{a}`	$\vec{a}$	`\vec{a}`		

B.2. Математические алфавиты

В табл. B.2 – B.7 приведены математические алфавиты. В колонках, помеченных значком •, приведены буквы, а в колонках с заголовком U+# указаны их шестнадцатеричные коды в кодировке unicode.

Команда `\mathit{•}` выводит буквы и цифры математического курсива, представленные в табл. В.2. Для вывода жирных символов используется команда `\bm{•}`. Цифры, отсутствующие в кодировке unicode, показаны без кодов.

При использовании пакета unicode-math математический курсив является основным шрифтом формул, поэтому команда `\mathit` для печати букв не требуется. Жирные буквы рекомендуется вводить командой `\symbfit{•}`.

Таблица В.2

Математический курсив

•	U+#	•	U+#	•	U+#	•	U+#	•
A	1d434	a	1d44e	$\boldsymbol{A}$	1d468	$\boldsymbol{a}$	1d482	
B	1d435	b	1d44f	$\boldsymbol{B}$	1d469	$\boldsymbol{b}$	1d483	
C	1d436	c	1d450	$\boldsymbol{C}$	1d46a	$\boldsymbol{c}$	1d484	0
D	1d437	d	1d451	$\boldsymbol{D}$	1d46b	$\boldsymbol{d}$	1d485	1
E	1d438	e	1d452	$\boldsymbol{E}$	1d46c	$\boldsymbol{e}$	1d486	2
F	1d439	f	1d453	$\boldsymbol{F}$	1d46d	$\boldsymbol{f}$	1d487	3
G	1d43a	g	1d454	$\boldsymbol{G}$	1d46e	$\boldsymbol{g}$	1d488	4
H	1d43b	h	1d455	$\boldsymbol{H}$	1d46f	$\boldsymbol{h}$	1d489	5
I	1d43c	i	1d456	$\boldsymbol{I}$	1d470	$\boldsymbol{i}$	1d48a	6
J	1d43d	j	1d457	$\boldsymbol{J}$	1d471	$\boldsymbol{j}$	1d48b	7
K	1d43e	k	1d458	$\boldsymbol{K}$	1d472	$\boldsymbol{k}$	1d48c	8
L	1d43f	l	1d459	$\boldsymbol{L}$	1d473	$\boldsymbol{l}$	1d48d	9
M	1d440	m	1d45a	$\boldsymbol{M}$	1d474	$\boldsymbol{m}$	1d48e	
N	1d441	n	1d45b	$\boldsymbol{N}$	1d475	$\boldsymbol{n}$	1d48f	
O	1d442	o	1d45c	$\boldsymbol{O}$	1d476	$\boldsymbol{o}$	1d490	$\boldsymbol{0}$
P	1d443	p	1d45d	$\boldsymbol{P}$	1d477	$\boldsymbol{p}$	1d491	$\boldsymbol{1}$
Q	1d444	q	1d45e	$\boldsymbol{Q}$	1d478	$\boldsymbol{q}$	1d492	$\boldsymbol{2}$
R	1d445	r	1d45f	$\boldsymbol{R}$	1d479	$\boldsymbol{r}$	1d493	$\boldsymbol{3}$
S	1d446	s	1d460	$\boldsymbol{S}$	1d47a	$\boldsymbol{s}$	1d494	$\boldsymbol{4}$
T	1d447	t	1d461	$\boldsymbol{T}$	1d47b	$\boldsymbol{t}$	1d495	$\boldsymbol{5}$
U	1d448	u	1d462	$\boldsymbol{U}$	1d47c	$\boldsymbol{u}$	1d496	$\boldsymbol{6}$
V	1d449	v	1d463	$\boldsymbol{V}$	1d47d	$\boldsymbol{v}$	1d497	$\boldsymbol{7}$
W	1d44a	w	1d464	$\boldsymbol{W}$	1d47e	$\boldsymbol{w}$	1d998	$\boldsymbol{8}$
X	1d44b	x	1d465	$\boldsymbol{X}$	1d47f	$\boldsymbol{x}$	1d499	$\boldsymbol{9}$
Y	1d44c	y	1d466	$\boldsymbol{Y}$	1d480	$\boldsymbol{y}$	1d49a	
Z	1d44d	z	1d467	$\boldsymbol{Z}$	1d481	$\boldsymbol{z}$	1d49b	

Команда `\mathrm{•}` выводит буквы и цифры математического шрифта прямого начертания, представленные в табл. B.3. Для вывода жирных символов используется команда `\mathbf{•}`.

Пакет unicode-math вводит дополнительные команды печати символов: `\symup{•}` и `\symbfup{•}`.

Таблица B.3

Прямой математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#	•	U+#
A	41	a	61	**A**	1d400	**a**	1d41a		
B	42	b	62	**B**	1d401	**b**	1d41b		
C	43	c	63	**C**	1d402	**c**	1d41c	0	30
D	44	d	64	**D**	1d403	**d**	1d41d	1	31
E	45	e	65	**E**	1d404	**e**	1d41e	2	32
F	46	f	66	**F**	1d405	**f**	1d41f	3	33
G	47	g	67	**G**	1d406	**g**	1d420	4	34
H	48	h	68	**H**	1d407	**h**	1d421	5	35
I	49	i	69	**I**	1d408	**i**	1d422	6	36
J	4a	j	6a	**J**	1d409	**j**	1d423	7	37
K	4b	k	6b	**K**	1d40a	**k**	1d424	8	38
L	4c	l	6c	**L**	1d40b	**l**	1d425	9	39
M	4d	m	6d	**M**	1d40c	**m**	1d426		
N	4e	n	6e	**N**	1d40d	**n**	1d427		
O	4f	o	6f	**O**	1d40e	**o**	1d428	**0**	1d7ce
P	50	p	70	**P**	1d40f	**p**	1d429	**1**	1d7cf
Q	51	q	71	**Q**	1d410	**q**	1d42a	**2**	1d7d0
R	52	r	72	**R**	1d411	**r**	1d42b	**3**	1d7d1
S	53	s	73	**S**	1d412	**s**	1d42c	**4**	1d7d2
T	54	t	74	**T**	1d413	**t**	1d42d	**5**	1d7d3
U	55	u	75	**U**	1d414	**u**	1d42e	**6**	1d7d4
V	56	v	76	**V**	1d415	**v**	1d42f	**7**	1d7d5
W	57	w	77	**W**	1d416	**w**	1d430	**8**	1d7d6
X	58	x	78	**X**	1d417	**x**	1d431	**9**	1d7d7
Y	59	y	79	**Y**	1d418	**y**	1d432		
Z	5a	z	7a	**Z**	1d419	**z**	1d433		

Команда `\mathsf{•}` выводит буквы и цифры математического рубленого шрифта, представленные в табл. B.4. Жирные символы печатают команды `\bm{\mathsf{•}}`.

Таблица B.4

Рубленый математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#	•	U+#
A	1d5a0	a	1d5ba	**A**	1d5d4	**a**	1d5ee		
B	1d5a1	b	1d5bb	**B**	1d5d5	**b**	1d5ef		
C	1d5a2	c	1d5bc	**C**	1d5d6	**c**	1d5f0	0	1d7e2
D	1d5a3	d	1d5bd	**D**	1d5d7	**d**	1d5f1	1	1d7e3
E	1d5a4	e	1d5be	**E**	1d5d8	**e**	1d5f2	2	1d7e4
F	1d5a5	f	1d5bf	**F**	1d5d9	**f**	1d5f3	3	1d7e5
G	1d5a6	g	1d5c0	**G**	1d5da	**g**	1d5f4	4	1d7e6
H	1d5a7	h	1d5c1	**H**	1d5db	**h**	1d5f5	5	1d7e7
I	1d5a8	i	1d5c2	**I**	1d5dc	**i**	1d5f6	6	1d7e8
J	1d5a9	j	1d5c3	**J**	1d5dd	**j**	1d5f7	7	1d7e9
K	1d5aa	k	1d5c4	**K**	1d5de	**k**	1d5f8	8	1d7ea
L	1d5ab	l	1d5c5	**L**	1d5df	**l**	1d5f9	9	1d7eb
M	1d5ac	m	1d5c6	**M**	1d5e1	**m**	1d5fa		
N	1d5ad	n	1d5c7	**N**	1d5e2	**n**	1d5fb		
O	1d5ae	o	1d5c8	**O**	1d5e3	**o**	1d5fc	**0**	1d7ec
P	1d5af	p	1d5c9	**P**	1d5e4	**p**	1d5fd	**1**	1d7ed
Q	1d5b0	q	1d5ca	**Q**	1d5e5	**q**	1d5fe	**2**	1d7ee
R	1d5b1	r	1d5cb	**R**	1d5e6	**r**	1d5ff	**3**	1d7ef
S	1d5b2	s	1d5cc	**S**	1d5e7	**s**	1d600	**4**	1d7f0
T	1d5b3	t	1d5cd	**T**	1d5e8	**t**	1d601	**5**	1d7f1
U	1d5b4	u	1d5ce	**U**	1d5e9	**u**	1d602	**6**	1d7f2
V	1d5b5	v	1d5cf	**V**	1d5ea	**v**	1d603	**7**	1d7f3
W	1d5b6	w	1d5d0	**W**	1d5ea	**w**	1d604	**8**	1d7f4
X	1d5b7	x	1d5d1	**X**	1d5eb	**x**	1d605	**9**	1d7f5
Y	1d5b8	y	1d5d2	**Y**	1d5ec	**y**	1d606		
Z	1d5b9	z	1d5d3	**Z**	1d5ed	**z**	1d607		
α	1d770	**β**	1d771	**γ**	1d772	**δ**	1d773	**ε**	1d774
ϵ	1d78a	**ζ**	1d775	**η**	1d776	**θ**	1d777	**ϑ**	1d78b
ι	1d778	**κ**	1d779	**ϰ**	1d78c	**λ**	1d77q	**μ**	1d77b
ν	1d77c	**ξ**	1d77d	**o**	1d77e	**π**	1d77f	**ϖ**	1d78f
ρ	1d780	**ϱ**	1d78e	**ς**	1d781	**σ**	1d782	**τ**	1d783
υ	1d784	**ϕ**	1d78d	**φ**	1d785	**χ**	1d786	**ψ**	1d787
ω	1d788	**Γ**	1d758	**Δ**	1d759	**Θ**	1d75d	**Θ**	1d767
Λ	1d760	**Ξ**	1d763	**Π**	1d765	**Σ**	1d768	**Υ**	1d76a
Φ	1d76b	**Ψ**	1d76d	**Ω**	1d76e	**∂**	1d789	**∇**	1d76f

В OpenType гарнитурах помимо латинских букв и цифр доступны жирные греческие буквы, а также рубленый математический курсив, представленный в табл. B.5. При использовании пакета unicode-math символы нормальной насыщенности печатают команды `\mathsf{•}` или `\symsfup{•}` и `\symsfit{•}` или `\mathsfit{•}`, а жирные — команды `\symbfsfup{•}` или `\mathbfsf{•}` и `\symbfsfit{•}` или `\mathbfsfit{•}`. Например, `\symbfsfit{aZ\alpha\Omega}` ⇝ ***aZαΩ***.

Таблица B.5

Рубленый математический курсив

•	U+#	•	U+#	•	U+#	•	U+#	•	U+#	•	U+#
A	1d608	*a*	1d622	***A***	1d63c	***a***	1d656	***α***	1d7aa	***φ***	1d7bf
B	1d609	*b*	1d623	***B***	1d63d	***b***	1d657	***β***	1d7ab	***ϕ***	1d7c7
C	1d60a	*c*	1d624	***C***	1d63e	***c***	1d658	***γ***	1d7ac	***χ***	1d7c0
D	1d60b	*d*	1d625	***D***	1d63f	***d***	1d659	***δ***	1d7ad	***ψ***	1d7c1
E	1d60c	*e*	1d626	***E***	1d640	***e***	1d65a	***ε***	1d7ae	***ω***	1d7c2
F	1d60d	*f*	1d627	***F***	1d641	***f***	1d65b	***ϵ***	1d7c4	***Γ***	1d792
G	1d60e	*g*	1d628	***G***	1d642	***g***	1d65c	***ζ***	1d7af	***Δ***	1d793
H	1d60f	*h*	1d629	***H***	1d643	***h***	1d65d	***η***	1d7b0	***Θ***	1d797
I	1d610	*i*	1d62a	***I***	1d644	***i***	1d65e	***θ***	1d7b1	***Θ***	1d7a1
J	1d611	*j*	1d62b	***J***	1d645	***j***	1d65f	***ϑ***	1d7c5	***Λ***	1d79a
K	1d612	*k*	1d62c	***K***	1d646	***k***	1d660	***ι***	1d7b2	***Ξ***	1d79d
L	1d613	*l*	1d62d	***L***	1d647	***l***	1d661	***κ***	1d7c6	***Π***	1d79f
M	1d614	*m*	1d62e	***M***	1d648	***m***	1d662	***κ***	1d7b3	***Σ***	1d7a2
N	1d615	*n*	1d62f	***N***	1d649	***n***	1d663	***λ***	1d7b4	***Υ***	1d7a4
O	1d616	*o*	1d630	***O***	1d64a	***o***	1d664	***μ***	1d7b5	***Φ***	1d7a5
P	1d617	*p*	1d631	***P***	1d64b	***p***	1d665	***ν***	1d7b6	***Ψ***	1d7a7
Q	1d618	*q*	1d632	***Q***	1d64c	***q***	1d666	***ξ***	1d7b7	***Ω***	1d7a8
R	1d619	*r*	1d633	***R***	1d64d	***r***	1d667	***o***	1d7b8	***∂***	1d7c3
S	1d61a	*s*	1d634	***S***	1d64e	***s***	1d668	***π***	1d7b9	***∇***	1d7a9
T	1d61b	*t*	1d635	***T***	1d64f	***t***	1d669	***ϖ***	1d7c9		
U	1d61c	*u*	1d636	***U***	1d650	***u***	1d66a	***ρ***	1d7ba		
V	1d61d	*v*	1d637	***V***	1d651	***v***	1d66b	***ϱ***	1d7c8		
W	1d61e	*w*	1d638	***W***	1d652	***w***	1d66c	***ς***	1d7bb		
X	1d61f	*x*	1d639	***X***	1d653	***x***	1d66d	***σ***	1d7bc		
Y	1d620	*y*	1d63a	***Y***	1d654	***y***	1d66e	***τ***	1d7bd		
Z	1d621	*z*	1d63b	***Z***	1d655	***z***	1d66f	***υ***	1d7be		

Команда `\mathtt{•}` выводит представленные в табл. B.6 буквы и цифры математического моноширинного шрифта нормальной насыщенности. Символы ажурного шрифта коллекции пакетов $\mathcal{AMS}$-LaTeX, представленные в табл. B.7, печатает команда `\mathbb{•}`. Строчные буквы и цифры имеются только в OpenType гарнитурах.

Таблица B.6

Моноширинный математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#	•	U+#
A	1d670	a	1d68a	N	1d67d	n	1d697	0	1d7f6
B	1d671	b	1d68b	O	1d67e	o	1d698	1	1d7f7
C	1d672	c	1d68c	P	1d67f	p	1d699	2	1d7f8
D	1d673	d	1d68d	Q	1d680	q	1d69a	3	1d7f9
E	1d674	e	1d68e	R	1d681	r	1d69b	4	1d7fa
F	1d675	f	1d68f	S	1d682	s	1d69c	5	1d7fb
G	1d676	g	1d690	T	1d683	t	1d69d	6	1d7fc
H	1d677	h	1d691	U	1d684	u	1d69e	7	1d7fd
I	1d678	i	1d692	V	1d685	v	1d69f	8	1d7fe
J	1d693	j	1d679	W	1d686	w	1d6a0	9	1d7ff
K	1d67a	k	1d694	X	1d687	x	1d6a1		
L	1d67b	l	1d695	Y	1d688	y	1d6a2		
M	1d67c	m	1d696	Z	1d689	z	1d6a3		

Таблица B.7

Ажурный математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#	•	U+#
𝔸	1d538	𝕒	1d552	ℕ	2115	𝕟	1d55f	𝟘	1d7d8
𝔹	1d539	𝕓	1d553	𝕆	1d546	𝕠	1d560	𝟙	1d7d9
ℂ	2102	𝕔	1d554	ℙ	2119	𝕡	1d561	𝟚	1d7da
𝔻	1d53b	𝕕	1d555	ℚ	211a	𝕢	1d562	𝟛	1d7db
𝔼	1d53c	𝕖	1d556	ℝ	211d	𝕣	1d563	𝟜	1d7dc
𝔽	1d53d	𝕗	1d557	𝕊	1d54a	𝕤	1d564	𝟝	1d7dd
𝔾	1d53e	𝕘	1d558	𝕋	1d54b	𝕥	1d565	𝟞	1d7de
ℍ	210d	𝕙	1d559	𝕌	1d54c	𝕦	1d566	𝟟	1d7df
𝕀	1d540	𝕚	1d55a	𝕍	1d54d	𝕧	1d567	𝟠	1d7e0
𝕁	1d541	𝕛	1d55b	𝕎	1d54e	𝕨	1d568	𝟡	1d7e1
𝕂	1d542	𝕜	1d55c	𝕏	1d54f	𝕩	1d569		
𝕃	1d543	𝕝	1d55d	𝕐	1d550	𝕪	1d56a		
𝕄	1d544	𝕞	1d55e	ℤ	2124	𝕫	1d56b		

Пакет eucal из коллекции пакетов $\mathcal{AMS}$-LaTeX подключает рукописный шрифт. При загрузке с параметром mathscr для печати букв вводится команда `\mathscr{•}`. Жирные буквы выводят команды `\bm{\mathscr{•}}`.

Рукописный шрифт $\mathcal{AMS}$ содержит только прописные буквы, а OpenType гарнитуры имеют также и строчные (см. табл. B.8). При использовании пакета unicode-math буквы печатают команды `\mathscr{•}` или `\symscr{•}` и `\symbfscr{•}` или `\mathbfscr{•}`.

Таблица B.8

Рукописный математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#
$\mathscr{A}$	1d49c	$\mathscr{a}$	1d4b6	$\boldsymbol{\mathscr{A}}$	1d4d0	$\boldsymbol{\mathscr{a}}$	1d4ea
$\mathscr{B}$	212c	$\mathscr{b}$	1d4b7	$\boldsymbol{\mathscr{B}}$	1d4d1	$\boldsymbol{\mathscr{b}}$	1d4eb
$\mathscr{C}$	1d49e	$\mathscr{c}$	1d4b8	$\boldsymbol{\mathscr{C}}$	1d4d2	$\boldsymbol{\mathscr{c}}$	1d4ec
$\mathscr{D}$	1d49f	$\mathscr{d}$	1d4b9	$\boldsymbol{\mathscr{D}}$	1d4d3	$\boldsymbol{\mathscr{d}}$	1d4ed
$\mathscr{E}$	2130	$\mathscr{e}$	1d4ba	$\boldsymbol{\mathscr{E}}$	1d4d4	$\boldsymbol{\mathscr{e}}$	1d4ee
$\mathscr{F}$	2131	$\mathscr{f}$	1d4bb	$\boldsymbol{\mathscr{F}}$	1d4d5	$\boldsymbol{\mathscr{f}}$	1d4ef
$\mathscr{G}$	1d4a2	$\mathscr{g}$	1d4bc	$\boldsymbol{\mathscr{G}}$	1d4d6	$\boldsymbol{\mathscr{g}}$	1d4f0
$\mathscr{H}$	210b	$\mathscr{h}$	1d4bd	$\boldsymbol{\mathscr{H}}$	1d4d7	$\boldsymbol{\mathscr{h}}$	1d4f1
$\mathscr{I}$	2110	$\mathscr{i}$	1d4be	$\boldsymbol{\mathscr{I}}$	1d4d8	$\boldsymbol{\mathscr{i}}$	1d4f2
$\mathscr{J}$	1d4a5	$\mathscr{j}$	1d4bf	$\boldsymbol{\mathscr{J}}$	1d4d9	$\boldsymbol{\mathscr{j}}$	1d4f3
$\mathscr{K}$	1d4a6	$\mathscr{k}$	1d4c0	$\boldsymbol{\mathscr{K}}$	1d4da	$\boldsymbol{\mathscr{k}}$	1d4f4
$\mathscr{L}$	2112	$\mathscr{l}$	1d4c1	$\boldsymbol{\mathscr{L}}$	1d4db	$\boldsymbol{\mathscr{l}}$	1d4f5
$\mathscr{M}$	2133	$\mathscr{m}$	1d4c2	$\boldsymbol{\mathscr{M}}$	1d4dc	$\boldsymbol{\mathscr{m}}$	1d4f6
$\mathscr{N}$	1d4a9	$\mathscr{n}$	1d4c3	$\boldsymbol{\mathscr{N}}$	1d4dd	$\boldsymbol{\mathscr{n}}$	1d4f7
$\mathscr{O}$	1d4aa	$\mathscr{o}$	1d4c4	$\boldsymbol{\mathscr{O}}$	1d4de	$\boldsymbol{\mathscr{o}}$	1d4f8
$\mathscr{P}$	1d4ab	$\mathscr{p}$	1d4c5	$\boldsymbol{\mathscr{P}}$	1d4df	$\boldsymbol{\mathscr{p}}$	1d4f9
$\mathscr{Q}$	1d4ac	$\mathscr{q}$	1d4c6	$\boldsymbol{\mathscr{Q}}$	1d4e0	$\boldsymbol{\mathscr{q}}$	1d4fa
$\mathscr{R}$	211b	$\mathscr{r}$	1d4c7	$\boldsymbol{\mathscr{R}}$	1d4e1	$\boldsymbol{\mathscr{r}}$	1d4fb
$\mathscr{S}$	1d4ae	$\mathscr{s}$	1d4c8	$\boldsymbol{\mathscr{S}}$	1d4e2	$\boldsymbol{\mathscr{s}}$	1d4fc
$\mathscr{T}$	1d4af	$\mathscr{t}$	1d4c9	$\boldsymbol{\mathscr{T}}$	1d4e3	$\boldsymbol{\mathscr{t}}$	1d4fd
$\mathscr{U}$	1d4b0	$\mathscr{u}$	1d4ca	$\boldsymbol{\mathscr{U}}$	1d4e4	$\boldsymbol{\mathscr{u}}$	1d4fe
$\mathscr{V}$	1d4b1	$\mathscr{v}$	1d4cb	$\boldsymbol{\mathscr{V}}$	1d4e5	$\boldsymbol{\mathscr{v}}$	1d4ff
$\mathscr{W}$	1d4b2	$\mathscr{w}$	1d4cc	$\boldsymbol{\mathscr{W}}$	1d4e6	$\boldsymbol{\mathscr{w}}$	1d500
$\mathscr{X}$	1d4b3	$\mathscr{x}$	1d4cd	$\boldsymbol{\mathscr{X}}$	1d4e7	$\boldsymbol{\mathscr{x}}$	1d501
$\mathscr{Y}$	1d4b4	$\mathscr{y}$	1d4ce	$\boldsymbol{\mathscr{Y}}$	1d4e8	$\boldsymbol{\mathscr{y}}$	1d502
$\mathscr{Z}$	1d4b5	$\mathscr{z}$	1d4cf	$\boldsymbol{\mathscr{Z}}$	1d4e9	$\boldsymbol{\mathscr{z}}$	1d503

Команда `\mathcal{•}` выводит буквы каллиграфического шрифта, представленные в табл. B.9. Жирные буквы печатают команды `\bm{\mathcal{•}}`. Шрифты Computer Modern содержат только прописные буквы.

OpenType гарнитуры обычно содержат только рукописные шрифты, которые подставляются вместо каллиграфических. Пакет unicode-math позволяет подгрузить «самостоятельный» каллиграфический шрифт из другой гарнитуры. Его буквы печатают команды `\mathcal{•}` или `\symcal{•}` и `\symbfcal{•}` или `\mathbfcal{•}`.

Таблица B.9

Каллиграфический математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#
$\mathcal{A}$	1d49c	$\mathcal{a}$	1d4b6	$\boldsymbol{\mathcal{A}}$	1d4d0	$\boldsymbol{\mathcal{a}}$	1d4ea
$\mathcal{B}$	212c	$\mathcal{b}$	1d4b7	$\boldsymbol{\mathcal{B}}$	1d4d1	$\boldsymbol{\mathcal{b}}$	1d4eb
$\mathcal{C}$	1d49e	$\mathcal{c}$	1d4b8	$\boldsymbol{\mathcal{C}}$	1d4d2	$\boldsymbol{\mathcal{c}}$	1d4ec
$\mathcal{D}$	1d49f	$\mathcal{d}$	1d4b9	$\boldsymbol{\mathcal{D}}$	1d4d3	$\boldsymbol{\mathcal{d}}$	1d4ed
$\mathcal{E}$	2130	$\mathcal{e}$	1d4ba	$\boldsymbol{\mathcal{E}}$	1d4d4	$\boldsymbol{\mathcal{e}}$	1d4ee
$\mathcal{F}$	2131	$\mathcal{f}$	1d4bb	$\boldsymbol{\mathcal{F}}$	1d4d5	$\boldsymbol{\mathcal{f}}$	1d4ef
$\mathcal{G}$	1d4a2	$\mathcal{g}$	1d4bc	$\boldsymbol{\mathcal{G}}$	1d4d6	$\boldsymbol{\mathcal{g}}$	1d4f0
$\mathcal{H}$	210b	$\mathcal{h}$	1d4bd	$\boldsymbol{\mathcal{H}}$	1d4d7	$\boldsymbol{\mathcal{h}}$	1d4f1
$\mathcal{I}$	2110	$\mathcal{i}$	1d4be	$\boldsymbol{\mathcal{I}}$	1d4d8	$\boldsymbol{\mathcal{i}}$	1d4f2
$\mathcal{J}$	1d4a5	$\mathcal{j}$	1d4bf	$\boldsymbol{\mathcal{J}}$	1d4d9	$\boldsymbol{\mathcal{j}}$	1d4f3
$\mathcal{K}$	1d4a6	$\mathcal{k}$	1d4c0	$\boldsymbol{\mathcal{K}}$	1d4da	$\boldsymbol{\mathcal{k}}$	1d4f4
$\mathcal{L}$	2112	$\mathcal{l}$	1d4c1	$\boldsymbol{\mathcal{L}}$	1d4db	$\boldsymbol{\mathcal{l}}$	1d4f5
$\mathcal{M}$	2133	$\mathcal{m}$	1d4c2	$\boldsymbol{\mathcal{M}}$	1d4dc	$\boldsymbol{\mathcal{m}}$	1d4f6
$\mathcal{N}$	1d4a9	$\mathcal{n}$	1d4c3	$\boldsymbol{\mathcal{N}}$	1d4dd	$\boldsymbol{\mathcal{n}}$	1d4f7
$\mathcal{O}$	1d4aa	$\mathcal{o}$	1d4c4	$\boldsymbol{\mathcal{O}}$	1d4de	$\boldsymbol{\mathcal{o}}$	1d4f8
$\mathcal{P}$	1d4ab	$\mathcal{p}$	1d4c5	$\boldsymbol{\mathcal{P}}$	1d4df	$\boldsymbol{\mathcal{p}}$	1d4f9
$\mathcal{Q}$	1d4ac	$\mathcal{q}$	1d4c6	$\boldsymbol{\mathcal{Q}}$	1d4e0	$\boldsymbol{\mathcal{q}}$	1d4fa
$\mathcal{R}$	211b	$\mathcal{r}$	1d4c7	$\boldsymbol{\mathcal{R}}$	1d4e1	$\boldsymbol{\mathcal{r}}$	1d4fb
$\mathcal{S}$	1d4ae	$\mathcal{s}$	1d4c8	$\boldsymbol{\mathcal{S}}$	1d4e2	$\boldsymbol{\mathcal{s}}$	1d4fc
$\mathcal{T}$	1d4af	$\mathcal{t}$	1d4c9	$\boldsymbol{\mathcal{T}}$	1d4e3	$\boldsymbol{\mathcal{t}}$	1d4fd
$\mathcal{U}$	1d4b0	$\mathcal{u}$	1d4ca	$\boldsymbol{\mathcal{U}}$	1d4e4	$\boldsymbol{\mathcal{u}}$	1d4fe
$\mathcal{V}$	1d4b1	$\mathcal{v}$	1d4cb	$\boldsymbol{\mathcal{V}}$	1d4e5	$\boldsymbol{\mathcal{v}}$	1d4ff
$\mathcal{W}$	1d4b2	$\mathcal{w}$	1d4cc	$\boldsymbol{\mathcal{W}}$	1d4e6	$\boldsymbol{\mathcal{w}}$	1d500
$\mathcal{X}$	1d4b3	$\mathcal{x}$	1d4cd	$\boldsymbol{\mathcal{X}}$	1d4e7	$\boldsymbol{\mathcal{x}}$	1d501
$\mathcal{Y}$	1d4b4	$\mathcal{y}$	1d4ce	$\boldsymbol{\mathcal{Y}}$	1d4e8	$\boldsymbol{\mathcal{y}}$	1d502
$\mathcal{Z}$	1d4b5	$\mathcal{z}$	1d4cf	$\boldsymbol{\mathcal{Z}}$	1d4e9	$\boldsymbol{\mathcal{z}}$	1d503

Коллекция пакетов $\mathcal{AMS}$-LaTeX содержит готический математический шрифт, представленный в табл. B.10. Для печати букв и цифр используются команды `\mathfrak{•}` и `\bm{\mathfrak{•}}`.

В OpenType гарнитурах цифры отсутствуют. При использовании пакета unicode-math буквы печатают команды `\mathfrak{•}` или `\symfrak{•}` и `\symbffrak{•}` или `\mathbffrak{•}`.

Таблица B.10

Готический математический шрифт

•	U+#	•	U+#	•	U+#	•	U+#
𝔄	1d504	𝔞	1d51e	𝕬	1d56c	𝖆	1d586
𝔅	1d505	𝔟	1d51f	𝕭	1d56d	𝖇	1d587
ℭ	1d520	𝔠	212d	𝕮	1d56e	𝖈	1d588
𝔇	1d507	𝔡	1d521	𝕯	1d56f	𝖉	1d589
𝔈	1d508	𝔢	1d522	𝕰	1d570	𝖊	1d58a
𝔉	1d509	𝔣	1d523	𝕱	1d571	𝖋	1d58b
𝔊	1d50a	𝔤	1d524	𝕲	1d571	𝖌	1d58c
ℌ	210c	𝔥	1d525	𝕳	1d573	𝖍	1d58d
ℑ	2111	𝔦	1d526	𝕴	1d574	𝖎	1d58e
𝔍	1d50d	𝔧	1d527	𝕵	1d575	𝖏	1d58f
𝔎	1d50e	𝔨	1d528	𝕶	1d576	𝖐	1d590
𝔏	1d50f	𝔩	1d529	𝕷	1d577	𝖑	1d591
𝔐	1d510	𝔪	1d52a	𝕸	1d578	𝖒	1d592
𝔑	1d511	𝔫	1d52b	𝕹	1d579	𝖓	1d593
𝔒	1d512	𝔬	1d52c	𝕺	1d57a	𝖔	1d594
𝔓	1d513	𝔭	1d52d	𝕻	1d57b	𝖕	1d595
𝔔	1d514	𝔮	1d52e	𝕼	1d57c	𝖖	1d596
ℜ	1d515	𝔯	1d52f	𝕽	1d57d	𝖗	1d597
𝔖	1d516	𝔰	1d530	𝕾	1d57e	𝖘	1d598
𝔗	1d517	𝔱	1d531	𝕿	1d57f	𝖙	1d599
𝔘	1d518	𝔲	1d532	𝖀	1d580	𝖚	1d59a
𝔙	1d519	𝔳	1d533	𝖁	1d581	𝖛	1d59b
𝔚	1d51a	𝔴	1d534	𝖂	1d582	𝖜	1d59c
𝔛	1d51b	𝔵	1d535	𝖃	1d583	𝖝	1d59d
𝔜	1d51c	𝔶	1d536	𝖄	1d584	𝖞	1d59e
ℨ	2128	𝔷	1d537	𝖅	1d585	𝖟	1d59f

В.3. Греческие буквы

В табл. В.11 приведены греческие буквы, их коды в кодировке unicode и команды, используемые для их вывода. Команды, выделенные фоном, определены пакетом upgreek (см. с. 148).

Таблица В.11

Греческие буквы (\mathalpha)

•	U+#	•	U+#	Команды	•	U+#	•	U+#	Команды
α	1d6fc	$\boldsymbol{\alpha}$	1d736	\alpha	α	3b1	**α**	1d6c2	\upalpha
β	1d6fd	$\boldsymbol{\beta}$	1d737	\beta	β	3b2	**β**	1d6c3	\upbeta
γ	1d6fe	$\boldsymbol{\gamma}$	1d738	\gamma	γ	3b3	**γ**	1d6c4	\upgamma
δ	1d6ff	$\boldsymbol{\delta}$	1d739	\delta	δ	3b4	**δ**	1d6c5	\updelta
ε	1d700	$\boldsymbol{\varepsilon}$	1d73a	\varepsilon	ε	3b5	**ε**	1d6c6	\upvarepsilon
ϵ	1d716	$\boldsymbol{\epsilon}$	1d750	\epsilon	ϵ	3f5	**ϵ**	1d6dc	\upepsilon
ζ	1d701	$\boldsymbol{\zeta}$	1d73b	\zeta	ζ	3b6	**ζ**	1d6c7	\upzeta
η	1d702	$\boldsymbol{\eta}$	1d73c	\eta	η	3b7	**η**	1d6c8	\upeta
θ	1d703	$\boldsymbol{\theta}$	1d73d	\theta	θ	3b8	**θ**	1d6c9	\uptheta
ϑ	1d717	$\boldsymbol{\vartheta}$	1d751	\vartheta	ϑ	3d1	**ϑ**	1d6dd	\upvartheta
ι	1d704	$\boldsymbol{\iota}$	1d73e	\iota	ι	3b9	**ι**	1d6ca	\upiota
κ	1d705	$\boldsymbol{\kappa}$	1d73f	\kappa	κ	3ba	**κ**	1d6cb	\upkappa
$\varkappa$	1d718	$\boldsymbol{\varkappa}$	1d752	\varkappa	ϰ	3f0	**ϰ**	1d6de	\upvarkappa
λ	1d706	$\boldsymbol{\lambda}$	1d740	\lambda	λ	3bb	**λ**	1d6cc	\uplambda
μ	1d707	$\boldsymbol{\mu}$	1d741	\mu	μ	3bc	**μ**	1d6cd	\upmu
ν	1d708	$\boldsymbol{\nu}$	1d742	\nu	ν	3bd	**ν**	1d6ce	\upnu
ξ	1d709	$\boldsymbol{\xi}$	1d743	\xi	ξ	3be	**ξ**	1d6cf	\upxi
π	1d70b	$\boldsymbol{\pi}$	1d745	\pi	π	3c0	**π**	1d6d1	\uppi
ϖ	1d71b	$\boldsymbol{\varpi}$	1d755	\varpi	ϖ	3d6	**ϖ**	1d6e1	\upvarpi

Окончание табл. B.11

•	U+#	•	U+#	Команды	•	U+#	•	U+#	Команды	
ρ	1d70c	$\boldsymbol{\rho}$	1d746	\rho	ρ	3c1	**ρ**	1d6d2	\uprho	
ϱ	1d71a	$\boldsymbol{\varrho}$	1d754	\varrho	ϱ	3f1	**ϱ**	1d6e0	\upvarrho	
ς	1d70d	$\boldsymbol{\varsigma}$	1d747	\varsigma	ς	3c2	**ς**	1d6d3	\upvarsigma	
σ	1d70e	$\boldsymbol{\sigma}$	1d748	\sigma	σ	3c3	**σ**	1d6d4	\upsigma	
τ	1d70f	$\boldsymbol{\tau}$	1d749	\tau	τ	3c4	**τ**	1d6d5	\uptau	
υ	1d710	$\boldsymbol{\upsilon}$	1d74a	\upsilon	υ	3c5	**υ**	1d6d6	\upupsilon	
φ	1d711	$\boldsymbol{\varphi}$	1d74b	\varphi	φ	3c6	**φ**	1d6d7	\upvarphi	
ϕ	1d719	$\boldsymbol{\phi}$	1d753	\phi	ϕ	3d5	**ϕ**	1d6df	\upphi	
χ	1d712	$\boldsymbol{\chi}$	1d74c	\chi	χ	3c7	**χ**	1d6d8	\upchi	
ψ	1d713	$\boldsymbol{\psi}$	1d74d	\psi	ψ	3c8	**ψ**	1d6d9	\uppsi	
$\digamma$	3dd	$\boldsymbol{\digamma}$	1d7cb		F	3dc	**F**	1d7ca	\digamma	
Γ	1d6e4	$\boldsymbol{\Gamma}$	1d71e	\varGamma	Γ	393	**Γ**	1d6aa	\Gamma,	\Upgamma
Δ	1d6e5	$\boldsymbol{\Delta}$	1d71f	\varDelta	Δ	393	**Δ**	1d6ab	\Delta,	\Updelta
Θ	1d6e9	$\boldsymbol{\Theta}$	1d723	\varTheta	Θ	398	**Θ**	1d6af	\Theta,	\Uptheta
Λ	1d6ec	$\boldsymbol{\Lambda}$	1d726	\varLambda	Λ	39b	**Λ**	1d6b2	\Lambda,	\Uplambda
Ξ	1d6ef	$\boldsymbol{\Xi}$	1d729	\varXi	Ξ	39e	**Ξ**	1d6b5	\Xi,	\Upxi
Π	1d6f1	$\boldsymbol{\Pi}$	1d72b	\varPi	Π	3a0	**Π**	1d6b7	\Pi,	\Uppi
Σ	1d6f4	$\boldsymbol{\Sigma}$	1d72e	\varSigma	Σ	3a3	**Σ**	1d6ba	\Sigma,	\Upsigma
Υ	1d6f6	$\boldsymbol{\Upsilon}$	1d730	\varUpsilon	Υ	3a5	**Υ**	1d6bc	\Upsilon,	\Upupsilon
Φ	1d6f7	$\boldsymbol{\Phi}$	1d731	\varPhi	Φ	3a6	**Φ**	1d6bd	\Phi,	\Upphi
Ψ	1d6f9	$\boldsymbol{\Psi}$	1d733	\varPsi	Ψ	3a8	**Ψ**	1d6bf	\Psi,	\Uppsi
Ω	1d6fa	$\boldsymbol{\Omega}$	1d734	\varOmega	Ω	3a9	**Ω**	1d6c0	\Omega,	\Upomega

В.4. Математические символы

В табл. В.12 собраны большие операторы, относящиеся к категории `\mathop`. В таблице представлен их вид в строчных и вынесенных формулах. Оператор `\bigtimes` вводит пакет mathtools.

Большие операторы могут иметь верхний и нижний пределы (индексы), положение которых регулируют команды `\limits`, `\nolimits`. С помощью команды

`\sideset{`*левые индексы*`}{`*правые индексы*`}{`*оператор*`}`

пределы можно поставить как слева, так и справа.

Таблица В.12

Большие операторы

Символы	U+#	Команды	Символы	U+#	Команды
$\int$ $\int$	222b	`\int`	Σ $\sum$	2211	`\sum`
$\oint$ $\oint$	222e	`\oint`	$\bigodot$ $\bigodot$	2a00	`\bigodot`
$\iint$ $\iint$	222c	`\iint`	$\bigoplus$ $\bigoplus$	2a01	`\bigoplus`
$\iiint$ $\iiint$	222d	`\iiint`	$\bigotimes$ $\bigotimes$	2a02	`\bigotimes`
$\iiiint$ $\iiiint$	2a0c	`\iiiint`	$\times$ $\bigtimes$	2a09	`\bigtimes`
$\int\cdots\int$ $\int\cdots\int$		`\idotsint`	$\biguplus$ $\biguplus$	2a04	`\biguplus`
$\cap$ $\bigcap$	22c2	`\bigcap`	$\cup$ $\bigcup$	22c3	`\bigcup`
$\sqcap$ $\bigsqcap$	2a05	`\bigsqcap`	$\sqcup$ $\bigsqcup$	2a06	`\bigsqcup`
$\vee$ $\bigvee$	22c1	`\bigvee`	$\wedge$ $\bigwedge$	22c0	`\bigwedge`
$\prod$ $\prod$	220f	`\prod`	$\coprod$ $\coprod$	2210	`\coprod`

Табл. В.13 содержит разделители, размер которых может меняться в зависимости от высоты заключенного в них выражения. Левые и правые скобки, а также разделители с именами, начинающимися с букв `l` и `r`, относятся к категориям `\mathopen` и `\mathclose`. Они име-

ют несимметричную отбивку слева и справа. Остальные разделители относятся к категории \mathord.

Таблица B.13

Разделители

•	U+#	Команды	•	U+#	Команды
(	28	(	)	29	)
[	5b	[	]	5d	]
{	7b	\lbrace, \{	}	7d	\rbrace, \}
/	2f	/	\	5c	\backslash
⟨	27e8	\langle	⟩	27e9	\rangle
⌈	2308	\lceil	⌉	2309	\rceil
⌊	230a	\lfloor	⌋	230b	\rfloor
⌜	231c	\ulcorner	⌝	231d	\urcorner
⌞	231e	\llcorner	⌟	231f	\lrcorner
↑	2191	\uparrow	↓	2193	\downarrow
↕	2195	\updownarrow	⇳	21d5	\Updownarrow
⇓	21d3	\Downarrow	⇑	21d1	\Uparrow
|	7c	\vert, |	‖	2016	\Vert, \\|
|		\lvert	|		\rvert
‖		\lVert	‖		\rVert

В табл. B.14 – B.19 собраны различные символы, разбитые по категориям.

Таблица B.14

Буквенные символы (категория \mathord)

•	U+#	Команды	•	U+#	Команды	•	U+#	Команды
ℑ	2111	\Im	ℜ	211c	\Re	א	2135	\aleph
ℏ	210f	\hslash	ℏ		\hbar	ב	2136	\beth
𝚤	1d6a4	\imath	𝚥	1d6a5	\jmath	ג	2137	\gimel
𝕜	1d55c	\Bbbk	ℓ	2113	\ell	ד	2138	\daleth
⅁	2132	\Finv	⅁	2141	\Game	ð	f0	\eth
¥	a5	\yen	℧	2127	\mho	℘	2118	\wp

Таблица B.15

Различные символы (категория \mathord)

•	U+#	Команды	•	U+#	Команды
∞	221e	\infty	∇	2207	\nabla
∂	2202	\partial	∀	2200	\forall
╲		\diagdown	╱		\diagup
⊥	22a5	\bot	⊤	22a4	\top
∃	2203	\exists	∄	2204	\nexists
‵	2035	\backprime	′	2032	\prime, '
¬	ac	\neg, \lnot	∅	2205	\emptyset
∁	2201	\complement	⦸	29b0	\varnothing
∡	2221	\measuredangle	∠	2220	\angle
∢	2222	\sphericalangle	√	221a	\surd
△	25b3	\triangle	▽	25bf	\triangledown
▼	25be	\blacktriangledown	▲	25b4	\blacktriangle
□	25a1	\square, \Box	■	25a0	\blacksquare
◊	25ca	\lozenge	⧫	29eb	\blacklozenge
◇	2662	\diamondsuit	◇	25c7	\Diamond
∫		\smallint	♡	2661	\heartsuit
♣	2663	\clubsuit	♠	2660	\spadesuit
★	2605	\bigstar	✠	2720	\maltese
♯	266f	\sharp	♭	266d	\flat
♮	266e	\natural	✓	2713	\checkmark
®	ae	\circledR	Ⓢ	24c8	\circledS

Таблица B.16

Бинарные операторы (категория \mathbin)

•	U+#	Команды	•	U+#	Команды
±	b1	\pm	∓	2213	\mp
×	d7	\times	∔	2214	\dotplus
÷	f7	\div	⋇	22c7	\divideontimes
⋋	22cb	\leftthreetimes	⋌	22cc	\rightthreetimes
⋉	22c9	\ltimes	⋊	22ca	\rtimes
∖	2216	\smallsetminus	\	29f5	\setminus
⊎	228e	\uplus	⨿	2a3f	\amalg
∩	2229	\cap	∪	222a	\cup
⊓	2293	\sqcap	⊔	2294	\sqcup
⋒	22d2	\Cap, \doublecap	⋓	22d3	\Cup, \doublecup

Окончание табл. B.16

●	U+#	Команды	●	U+#	Команды
⋖	22d6	\lessdot	⋗	22d7	\gtrdot
⋎	22ce	\curlyvee	⋏	22cf	\curlywedge
∨	2228	\vee, \lor	∧	2227	\wedge, \lnot
⊻	22bb	\veebar	⊼	22bc	\barwedge
⊺	22ba	\intercal	⩞	2306	\doublebarwedge
†	2020	\dagger	‡	2021	\ddagger
★	22c6	\star	∗	204e	\ast
·	22c5	\cdot	▪	2b1d	\centerdot
⊠	22a0	\boxtimes	⊟	229f	\boxminus
⊞	229e	\boxplus	⊡	22a1	\boxdot
≀	2240	\wr	•	2219	\bullet
◯	25ef	\bigcirc	∘	2218	\circ
⊘	2298	\oslash	⊛	229b	\circledast
⊗	2297	\otimes	⊖	2296	\ominus
⊕	2295	\oplus	⊙	2299	\odot
⊖	229d	\circleddash	⊚	229a	\circledcirc
▽	25bd	\bigtriangledown	△	25b3	\bigtriangleup
◁	25c3	\triangleleft	▷	25b9	\triangleright
◁	25c1	\lhd	▷	25b7	\rhd
⊴	22b4	\unlhd	⊵	22b5	\unrhd
⋄	22c4	\diamond			

Таблица B.17

Соотношения (категория \mathrel)

●	U+#	Команды	●	U+#	Команды
≡	2261	\equiv	≅	2245	\cong
∝	221d	\propto	∝		\varpropto
∼	223c	\sim	≃	2243	\simeq
≈	2248	\approx	≋	224a	\approxeq
∽		\backsim	≌	22cd	\backsimeq
∼		\thicksim	≈		\thickapprox
≖	2256	\eqcirc	≗	2257	\circeq
≐	2250	\doteq	≑	2251	\doteqdot, \Doteq
≒	2252	\fallingdotseq	≓	2253	\risingdotseq
≎	224e	\Bumpeq	≏	224f	\bumpeq

Продолжение табл. B.17

•	U+#	Команды	•	U+#	Команды
$\in$	2208	\in	$\ni$	220b	\ni, \owns
$\ll$	226a	\ll	$\gg$	226b	\gg
$\lll$	22d8	\lll, \llless	$\ggg$	22d9	\ggg, \gggtr
$\leq$	2264	\leq, \le	$\geq$	2265	\geq, \ge
$\leqq$	2266	\leqq	$\geqq$	2267	\geqq
$\leqslant$	2a7d	\leqslant	$\geqslant$	2a7e	\geqslant
$\lesssim$	2a95	\eqslantless	$\gtrsim$	2a96	\eqslantgtr
$\lesssim$	2272	\lesssim	$\gtrsim$	2273	\gtrsim
$\lessapprox$	2a85	\lessapprox	$\gtrapprox$	2a86	\gtrapprox
$\lessgtr$	2276	\lessgtr	$\gtrless$	2277	\gtrless
$\lesseqgtr$	22da	\lesseqgtr	$\gtreqless$	22db	\gtreqless
$\lesseqqgtr$	2a8b	\lesseqqgtr	$\gtreqqless$	2a8c	\gtreqqless
$\subset$	2282	\subset	$\supset$	2283	\supset
$\subseteq$	2286	\subseteq	$\supseteq$	2287	\supseteq
$\subseteqq$	2ac5	\subseteqq	$\supseteqq$	2ac6	\supseteqq
$\Subset$	22d0	\Subset	$\Supset$	22d1	\Supset
$\sqsubset$	228f	\sqsubset	$\sqsupset$	2290	\sqsupset
$\sqsubseteq$	2291	\sqsubseteq	$\sqsupseteq$	2292	\sqsupseteq
$\prec$	227a	\prec	$\succ$	227b	\succ
$\preceq$	2aaf	\preceq	$\succeq$	2ab0	\succeq
$\preccurlyeq$	227c	\preccurlyeq	$\succcurlyeq$	227d	\succcurlyeq
$\curlyeqprec$	22de	\curlyeqprec	$\curlyeqsucc$	22df	\curlyeqsucc
$\precsim$	227e	\precsim	$\succsim$	227f	\succsim
$\precapprox$	2ab7	\precapprox	$\succapprox$	2ab8	\succapprox
$\vartriangleleft$	22b2	\vartriangleleft	$\vartriangleright$	22b3	\vartriangleright
$\trianglelefteq$	22b4	\trianglelefteq	$\trianglerighteq$	22b5	\trianglerighteq
$\blacktriangleleft$	25c2	\blacktriangleleft	$\blacktriangleright$	25b8	\blacktriangleright
$\bowtie$	22c8	\bowtie	$\Join$	2a1d	\Join
$\vartriangle$	25b5	\vartriangle	$\triangleq$	225c	\triangleq
$\therefore$	2234	\therefore	$\because$	2235	\because
$\frown$	2323	\frown	$\smile$	2322	\smile
$\smallfrown$		\smallfrown	$\smallsmile$		\smallsmile
$\between$	226c	\between	$\asymp$	224d	\asymp
$\mid$	2223	\mid	$\shortmid$		\shortmid
$\parallel$	2225	\parallel	$\shortparallel$		\shortparallel
$\vdash$	22a2	\vdash	$\dashv$	22a3	\dashv

Окончание табл. B.17

•	U+#	Команды	•	U+#	Команды
⊨	22a8	\models	⊦	22a7	\vDash
⊩	22a9	\Vdash	⊪	22aa	\Vvdash
⊥	27c2	\perp	⋔	22d4	\pitchfork
϶	3f6	\backepsilon			

Таблица B.18

Стрелки (соотношения, категория \mathrel)

•	U+#	Команды	•	U+#	Команды
←	2190	\leftarrow, \gets	→	2192	\rightarrow, \to
↚	219a	\nleftarrow	↛	219b	\nrightarrow
⟵	27f5	\longleftarrow	⟶	27f6	\longrightarrow
⇇	21c7	\leftleftarrows	⇆	21c6	\leftrightarrows
⇉	21c9	\rightrightarrows	⇄	21c4	\rightleftarrows
⇐	21d0	\Leftarrow	⇒	21d2	\Rightarrow
⇍	21cd	\nLeftarrow	⇏	21cf	\nRightarrow
⟸	27f8	\Longleftarrow	⟹	27f9	\Longrightarrow
⇚	21da	\Lleftarrow	⇛	21db	\Rrightarrow
↔	2194	\leftrightarrow	⇔	21d4	\Leftrightarrow
⟷	27f7	\longleftrightarrow	↮	21ae	\nleftrightarrow
⟺	27fa	\Longleftrightarrow	⇎	21ce	\nLeftrightarrow
↑	2191	\uparrow	↓	2193	\downarrow
↟		*nuparrow*	↡		*ndownarrow*
⇈	21c8	\upuparrows	⇊	21ca	\downdownarrows
⇑	21d1	\Uparrow	⇓	21d3	\Downarrow
↕	2195	\updownarrow	⇕	21d5	\Updownarrow
↗	2197	\nearrow	↘	2198	\searrow
↖	2196	\nwarrow	↙	2199	\swarrow
↿	21bf	\upharpoonleft	↾	21be	\upharpoonright \restriction
⇃	21c3	\downharpoonleft	⇂	21c2	\downharpoonright
↽	21bd	\leftharpoondown	⇁	21c1	\rightharpoondown
↼	21bc	\leftharpoonup	⇀	21c0	\rightharpoonup
⇋	21cb	\leftrightharpoons	⇌	21cc	\rightleftharpoons
↦	21a6	\mapsto	⟼	27fc	\longmapsto
⊸	22b8	\multimap	⇝	2933	\leadsto

Окончание табл. B.18

•	U+#	Команды	•	U+#	Команды
$\leftrightsquigarrow$	21ad	\leftrightsquigarrow	$\rightsquigarrow$	21dd	\rightsquigarrow
$\dashleftarrow$	21e0	\dashleftarrow	$\dashrightarrow$	21e2	\dashrightarrow
$\twoheadleftarrow$	219e	\twoheadleftarrow	$\twoheadrightarrow$	21a0	\twoheadrightarrow
$\leftarrowtail$	21a2	\leftarrowtail	$\rightarrowtail$	21a3	\rightarrowtail
$\hookleftarrow$	21a9	\hookleftarrow	$\hookrightarrow$	21aa	\hookrightarrow
$\looparrowleft$	21ab	\looparrowleft	$\looparrowright$	21ac	\looparrowright
$\curvearrowleft$	21b6	\curvearrowleft	$\curvearrowright$	21b7	\curvearrowright
$\circlearrowleft$	21ba	\circlearrowleft	$\circlearrowright$	21bb	\circlearrowright
$\Lsh$	21b0	\Lsh	$\Rsh$	21b1	\Rsh

Стрелки $\nuparrow$ (\nuparrow) и $\ndownarrow$ (\ndownarrow) вводит пакет mathtools.

Таблица B.19

Соотношения с отрицанием (категория \mathrel)

•	U+#	Команды	•	U+#	Команды
$\neq$	2260	\neq, \ne			
$\nsim$	2241	\nsim	$\ncong$	2245	\ncong
$\nless$	226e	\nless	$\ngtr$	226f	\ngtr
$\nleq$	2270	\nleq	$\ngeq$	2271	\ngeq
$\lneq$	2a87	\lneq	$\gneq$	2a88	\gneq
$\nleqslant$		\nleqslant	$\ngeqslant$		\ngeqslant
$\nleqq$		\nleqq	$\ngeqq$		\ngeqq
$\lneqq$	2268	\lneqq	$\gneqq$	2269	\gneqq
$\lvertneqq$		\lvertneqq	$\gvertneqq$		\gvertneqq
$\lnsim$	22e6	\lnsim	$\gnsim$	22e7	\gnsim
$\lnapprox$	2a89	\lnapprox	$\gnapprox$	2a8a	\gnapprox
$\nsubseteq$	2288	\nsubseteq	$\nsupseteq$	2289	\nsupseteq
$\subsetneq$	228a	\subsetneq	$\supsetneq$	228b	\supsetneq
$\varsubsetneq$		\varsubsetneq	$\varsupsetneq$		\varsupsetneq
$\nsubseteqq$		\nsubseteqq	$\nsupseteqq$		\nsupseteqq
$\subsetneqq$	2acb	\subsetneqq	$\supsetneqq$	2acc	\supsetneqq
$\varsubsetneqq$		\varsubsetneqq	$\varsupsetneqq$		\varsupsetneqq
$\nprec$	2280	\nprec	$\nsucc$	2281	\nsucc
$\npreceq$	22e0	\npreceq	$\nsucceq$	22e1	\nsucceq
$\precneqq$	2ab5	\precneqq	$\succneqq$	2ab6	\succneqq

Окончание табл. В.19

•	U+#	Команды	•	U+#	Команды
$\precnsim$	22e8	\precnsim	$\succnsim$	22e9	\succnsim
$\precnapprox$	2ab9	\precnapprox	$\succnapprox$	2aba	\succnapprox
$\ntriangleleft$	22ea	\ntriangleleft	$\ntriangleright$	22eb	\ntriangleright
$\ntrianglelefteq$	22ec	\ntrianglelefteq	$\ntrianglerighteq$	22ed	\ntrianglerighteq
$\nmid$	2224	\nmid	$\nparallel$	2226	\nparallel
$\nshortmid$		\nshortmid	$\nshortparallel$		\nshortparallel
$\nvdash$	22ac	\nvdash	$\nvDash$	22ad	\nvDash
$\nVdash$	22ae	\nVdash	$\nVDash$	22af	\nVDash

Отрицание отсутствующих в таблице отношений можно построить с помощью оператора \not, например \not\preccurlyeq $\nsim$.

В.5. Пределы и функции

В табл. В.20 и В.21 приведены функции и пределы, определенные при использовании пакета amsmath и загрузке пакета babel с настройками русского языка.

Таблица В.20

Математические функции

•	Команды	•	Команды	•	Команды
sin	\sin	arcsin	\arcsin	sec	\sec
cos	\cos	arccos	\arccos	csc	\csc
tan	\tan	arctan	\arctan	cot	\cot
tg	\tg	arctg	\arctg	cosec	\cosec
ctg	\ctg	arcctg	\arcctg		
sh	\sh	ch	\ch	th	\th
sinh	\sinh	cosh	\cosh	tanh	\tanh
exp	\exp	coth	\coth	cth	\cth
ln	\ln	lg	\lg	log	\log
arg	\arg	deg	\deg	dim	\dim
hom	\hom	ker	\ker	sup	\sup
P	\Prob	D	\Variance		
$y \bmod x$	y \bmod x	$y\ (x)$	y \pod x		
$y \mod x$	y \mod x	$y \pmod x$	y \pmod x		

Функции, представленные в табл. В.21, могут иметь верхние и нижние индексы, которые регулируются так же, как пределы больших операторов, например, `\displaystyle\det_a^b` ⇝ $\displaystyle\det_a^b$.

Таблица В.21

Функции, имеющие пределы

•	Команды	•	Команды	•	Команды	•	Команды
$\det$	`\det`	$\max$	`\max`	$\limsup$	`\limsup`	$\overline{\lim}$	`\varlimsup`
$\gcd$	`\gcd`	$\min$	`\min`	$\liminf$	`\liminf`	$\underline{\lim}$	`\varliminf`
$\inf$	`\inf`			$\varinjlim$	`\injlim`	$\varinjlim$	`\varinjlim`
$\Pr$	`\Pr`	$\lim$	`\lim`	$\varprojlim$	`\projlim`	$\varprojlim$	`\varprojlim`

С помощью команды `\operatorname{`*функция*`}` можно создать недостающую функцию непосредственно при ее использовании.

Декларации `\DeclareMathOperator{`*команда*`}{`*функция*`}` определяют команды для вывода новых функций.

Функции с пределами создают команды `\operatorname*{•}` и декларации `\DeclareMathOperator*{•}{•}`.

В.6. Точки, двоеточия и многоточия

Таблица В.22

Точки и многоточия (категория `\mathpunct`)

Примеры	U+#	Команды	Примеры	Команды
$x : x$	2236	`\colon`	$x.\,x$	`\ldotp`
$x \cdot x$	B7	`\cdot`	$x \cdot x$	`\cdotp`
$x \ldots x$	2026	`\ldots`	$0 \ldots 9$	`\dotso`
$x \cdots x$	22EF	`\cdots`	$+ \cdots$	`\dotsb`
$x \vdots x$	22EE	`\vdots`	$\times \cdots$	`\dotsm`
$x \ddots x$	22F1	`\ddots`	$x \ldots x$	`\dots`
$\int_b^a \cdots$		`\dotsi`	$, \ldots$	`\dotsc`

В табл. В.22 представлены различные виды точек и многоточий. Для символов кодировки unicode приведены шестнадцатеричные коды.

Символы \cdotp, \ldotp и \colon имеют правую отбивку. Отбивка обычного двоеточия симметрична, а точка отбивается слева. Сравните:

x.x ⇝ x.x и x.x ↜ x\ldotp x,

или

x:x ⇝ x : x и x: x ↜ x\colon x.

Многоточия \dotsb, \dotsc, \dotsi, \dotsm и \dotso предназначены для использования, соответственно, после бинарного оператора, запятой, интеграла, знака умножения и чего-то др.

Положение многоточия, заданного командой \dots, изменяется в зависимости от контекста выражения, следующего за ним.

В табл. B.23 собраны символы с двоеточием из пакета mathtools.

Таблица B.23

Символы с двоеточием (категория \mathrel)

Примеры	U+#	Команды	Примеры	U+#	Команды
$x : x$	2236	\colon	$x :: x$	2237	\dblcolon
$x :- x$		\colondash	$x -: x$	2239	\dashcolon
$x ::- x$		\Colondash	$x -:: x$		\Dashcolon
$x ≔ x$	2254	\coloneq	$x =: x$	2255	\eqcolon
$x ⩴ x$	2A74	\Coloneq	$x =:: x$		\Eqcolon
$x :\sim x$		\colonsim	$x \sim: x$		\simcolon
$x ::\sim x$		\Colonsim	$x \sim:: x$		\Simcolon
$x :\approx x$		\colonapprox	$x \approx: x$		\approxcolon
$x ::\approx x$		\Colonapprox	$x \approx:: x$		\Approxcolon

Приложение Г

Верстка по ГОСТу

Ряд российских ГОСТов регламентирует правила оформления научно-технической документации. Очень часто их необходимо соблюдать при оформлении отчетов о научно-исследовательской работе, авторефератов и диссертаций. Напрямую ни один стандартный класс LaTeX не обеспечивает выполнение требований ГОСТов, но их можно настроить для этого. В данном приложении рассмотрим настройку верстки отчетов и диссертаций. Изложение построено следующим образом. Сначала излагаются требования ГОСТов к отдельным элементам документов, а затем мелким шрифтом описываются настройки, обеспечивающие их соблюдение. При первом прочтении описания настроек можно опустить, вернувшись к ним, если возникнут проблемы с версткой документов по шаблону. В конце разделов приводятся макеты документов и собранные воедино настройки.

Отметим, что в данном приложении перечисления оформлены по ГОСТу, поэтому их формат отличается от перечней других глав.

Г.1. Отчет о НИР

Оформление отчетной научно-технической документации регламентирует ГОСТ 7.32-2017 «Отчет о научно-исследовательской работе. Структура и правила оформления» [46]. Стандарт распространяется на отчеты о фундаментальных, поисковых и прикладных научно-исследовательских работах по всем областям науки и техники.

Отчет о НИР выполняют любым печатным способом черным шрифтом на одной стороне листа белой бумаги формата А4 через полтора интервала, соблюдая следующие размеры полей: левое — 30 мм, правое — 15 мм, верхнее и нижнее — 20 мм. Заключительный отчет допускается печатать через один интервал, если его объем превышает 500 с. Абзацный отступ длиной 1.25 см должен быть одинаковым по всему тексту отчета. Рекомендуемый тип основного шрифта — Times New Roman размером не менее 12pt. Полужирный шрифт применяют только для заголовков разделов, подразделов и структурных элементов. Использование курсива допускается для обозначения объектов и печати терминов, в том числе терминов на латыни. Для акцентирования внимания текст можно выделять шрифтом иного начертания, но того же кегля и гарнитуры. Для оформления определенных терминов, формул и теорем разрешается применять шрифты разной гарнитуры.

Страницы отчета нумеруют арабскими цифрами, соблюдая сквозную нумерацию по всему тексту, включая приложения. Приложения, имеющие собственную нумерацию, можно не перенумеровать. Номер страницы проставляют в центре нижней части страницы без точки.

Структурными элементами отчета являются:
– **титульный лист**;
– **список исполнителей**;
– **реферат**;
– **содержание**;
– термины и определения;
– перечень сокращений и обозначений;
– **введение**;
– **основная часть отчета**;
– **заключение**;
– список использованных источников;
– приложения.

Обязательные элементы отчета выделены полужирным шрифтом, а остальные включают по усмотрению исполнителя.

Для верстки отчетов лучше всего подходят классы документа report или extreport, загружаемые с параметром a4paper и нужным кеглем. Требуемый стиль нумерации страниц соответствует их автоматическим настройкам. Подключение шрифтов Times New Roman обсуждалось в разд. 12.1.4. Межстрочный интервал можно установить декларацией `\onehalfspacing` в преамбуле рукописи, загрузив пакет setspace (см. с. 190).

Титульный лист

Титульный лист включают в общую нумерацию, но номер страницы на нем не проставляют. На титульном листе приводят следующие сведения:

а) наименование министерства (ведомства) или другого структурного образования, в систему которого входит организация-исполнитель (наименование приводят в полном виде с прописной буквы в верхней части титульного листа одно под другим по центру страницы);

б) полное и сокращенное наименование организации-исполнителя НИР (полное наименование приводят прописными буквами, по центру страницы, через один межстрочный интервал; сокращенное наименование приводят в круглых скобках, на отдельной строке, по центру страницы, прописными буквами, через один межстрочный интервал);

в) индекс Универсальной десятичной классификации (УДК) по ГОСТ Р 7.0.90-2016 [99], регистрационный номер НИР, регистрационный номер отчета (эти данные размещаются слева одно под другим, через один межстрочный интервал; от наименования организации индекс УДК отделяют двумя межстрочными интервалами);

г) грифы согласования и утверждения отчета, включая подпись руководителя организации с расшифровкой, печать организации и даты согласования и утверждения отчета (грифы состоят из слов: СОГЛАСОВАНО и УТВЕРЖДАЮ, наименования должности, ученой степени, ученого звания лица, согласовавшего и утвердившего отчет, личной подписи (синими чернилами), расшифровки подписи (инициалы и фамилия), даты согласования и утверждения отчета, здесь же проставляется печать организации, согласовавшей и утвердившей отчет; гриф СОГЛАСОВАНО размещается на титульном листе слева, а УТВЕРЖДАЮ — справа на два межстрочных интервала ниже регистрационного номера отчета; дата согласования и утверждения оформляется арабскими цифрами в последовательности: день (пара цифр), месяц (пара цифр), год (четыре цифры); допускается словесно-цифровой способ оформления даты; данные под грифами согласования и утверждения приводят через один межстрочный интервал);

д) вид документа (по центру страницы на первой строке приводят слово «ОТЧЕТ», на следующей строке — слова «О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ», через один межстрочный интервал, отступив от грифов согласования и утверждения два межстрочных интервала);

е) наименование НИР и отчета (каждое наименование приводят на отдельной строке по центру, через один межстрочный интервал; наименование НИР печатают строчными буквами с первой прописной, а наименование отчета – прописными буквами; если наименования различаются, между ними через один межстрочный интервал по центру приводят строчными буквами слова «по теме:»);

ж) вид отчета — заключительный, промежуточный (приводят в круглых скобках строчными буквами; для промежуточных отчетов в тех же круглых скобках, через запятую, указывают номер этапа);

з) номер (шифр) научно-технической программы, темы (номера печатают по центру страницы с прописной буквы, через один межстрочный интервал);

и) номер книги отчета (если отчет состоит из нескольких книг, то на титульном листе приводится слово «Книга» строчными буквами с первой прописной и через пробел номер текущей книги; каждая книга должна иметь свой титульный лист, соответствующий титульному листу первой книги и содержащий сведения, относящиеся к данной книге);

к) должность, ученая степень, ученое звание, подпись, инициалы и фамилия научного руководителя/руководителей НИР (должности, ученые степени, ученые звания указывают слева, затем оставляют свободное поле для подписей, справа указывают инициалы и фамилии; если на титульном листе не размещаются все необходимые подписи, то их переносят на дополнительную страницу, в правом верхнем углу которой указывают «Продолжение титульного листа», а в нижней части первой страницы справа — «Продолжение на следующем листе»);

л) место (город или другое место выполнения отчета) и год составления отчета (эти данные печатают по центру в нижней части титульного листа, отделяя друг от друга пробелом; если титульный лист имеет продолжение на следующем листе, то данные приводятся только на первом листе).

Перечисленные элементы титульного листа, кроме п. з и п. и, содержат обязательные сведения.

Верстка титульного листа уже обсуждалась ранее на с. 48. Хотя требования ГОСТа немного усложняют его структуру, показанную на рис. Г.1, его можно сформировать теми же методами:

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
(ФГАОУ ВО НИ ТПУ)

УДК 615.277.015:616-092.4
Рег. № НИОКТР 114011110269

УТВЕРЖДАЮ
Проректор по научной работе

_____ А.Н. Дьяченко
«____» _____ 2015 г.

ОТЧЕТ
О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ
Исследования радиофармацевтического препарата на основе
меченного технецием-99М доксорубицина
по теме:
ИССЛЕДОВАНИЕ МЕХАНИЗМА ДЕЙСТВИЯ РФМ
(промежуточный, этап 3)

Руководитель НИР,
зам. директора по научной работе

_____ В.С. Скуридин

Москва 2015

Рис. Г.1. Титульный листа отчета о НИР

```
\begin{center}
\singlespacing
Министерство образования и науки Российской Федерации\\
                          ...
\vspace{\baselineskip}
\begin{flushleft}
УДК 615.277.015:616-092.4\\
Рег. № НИОКТР 114011110269
\end{flushleft}

\vspace{\baselineskip}
\mbox{}\hfill
\begin{minipage}{.4\linewidth}
   УТВЕРЖДАЮ\\
   Проректор по научной работе\\[2ex]
   \rule{6em}{.4pt} А.Н. Дьяченко\\
   «\rule{2em}{.4pt}» \rule{5em}{.4pt} 2015 г.
\end{minipage}

\vfill
ОТЧЕТ\\
         ...
по теме:\\
         ...
(промежуточный, этап 3)

\vfill
\begin{flushleft}
   Руководитель НИР,\\
         ...
\end{flushleft}

\vfill
Москва 2015
\end{center}
```

Большинство элементов титульного листа центрируется, поэтому для верстки использовано окружение center, в начале которого декларация \singlespacing устанавливает одинарный межстрочный интервал. Команды \vspace и \vfill формируют вертикальную отбивку элементов.[1] Индекс

[1] Из примеров, приведенных в ГОСТ 7.32-2017, следует, что высоту отбивок можно увеличивать, поэтому часть команд \vspace заменена на \vfill.

УДК и подпись руководителя НИР помещены в окружения flushleft, выравнивающие текст слева. Гриф утверждения отчета помещен в министраницу, прижатую направо жесткой пружиной `\hfil`. В приведенном примере отсутствует гриф согласования, замененный пустым боксом `\mbox{}`, который, если потребуется, нужно заменить министраницей, указав в ней данные грифа. Команды `\rule` рисуют необходимые линии.

Список исполнителей

Список исполнителей должен содержать подписи руководителей НИР, ответственных исполнителей, исполнителей и соисполнителей, принимавших непосредственное участие в выполнении НИР. Исполнителей перечисляют столбцом в порядке должностей. Слева указывают их должности, ученые степени, ученые звания, затем оставляют свободное поле для подлинных подписей, справа указывают инициалы и фамилии. Возле каждой фамилии в скобках следует указать номер раздела (подраздела), в подготовке которого участвовал конкретный исполнитель. Для соисполнителей из других организаций нужно указать наименование организации-соисполнителя.

В конце списка расписывается номоконтролер, основная задача которого — проверка соблюдения норм и требований, установленных стандартом, соблюдение всех нормативных требований и единообразия в оформлении структурных элементов отчета.

Подписывая отчет, исполнители должны поставить собственной рукой дату.

Если отчет выполнен одним исполнителем, его должность, ученую степень, ученое звание, фамилию и инициалы указывают на титульном листе отчета, а список исполнителей не оформляют.

Список исполнителей можно сформировать в окружении tabular или longtable со следующей структурой:

```
\begin{tabular}{ >\raggedright b{.43\textwidth}
                 !{ \rule{.21\textwidth}{.4pt} }
                 >\raggedright p{.27\textwidth} }
  Руководитель НИР, \linebreak ... &
  А.Б. Кабакович \linebreak\footnotesize (...) \tabularnewline
```

Выбранный формат колонок обеспечит возможность ввода нескольких строк, а декларация `\raggedright` выровнит их слева. Команда, рисующая линии, помещена в разделитель `!{...}`. Ширина колонок и длина линий легко настраивается значениями аргументов `{...cm}`. Для перевода строк в колонках

используется команда `\linebreak`, а в таблице — `\tabularnewline`. Пример списка, сформированного описанным способом, показан на рис. Г.2.

СПИСОК ИСПОЛНИТЕЛЕЙ

Руководитель НИР, зам. директора по научной работе, д-р социол. наук	_____	А.Б. Кабакович (введение, заключение)
Отв. исполнитель, зав. отделом, канд. техн. наук	_____	С.П. Наливайко (раздел 1, заключение)
Исполнители: Ст. науч. сотр., канд. техн. наук	_____	А.В. Стопкин (раздел 1, 2, 3)
Нормоконтролер	_____	М.В. Закускина

Рис. Г.2. Список исполнителей отчета о НИР

Реферат

Реферат должен содержать:

– сведения об общем объеме отчета, количестве книг, иллюстраций, таблиц, использованных источников и приложений (сведения располагаются с абзацного отступа, в строку, через запятые);

– перечень ключевых слов, включающий от пяти до пятнадцати слов или словосочетаний из текста отчета, которые в наибольшей мере характеризуют его содержание и обеспечивают возможность информационного поиска (слова приводятся в именительном падеже и печатаются прописными буквами, в строку, через запятые, без абзацного отступа и переносов, без точки в конце перечня);

– текст реферата, который должен отражать:

а) объект исследования или разработки;

б) цель работы;

в) методы или методологию проведения работы;

г) результаты работы и их новизну;

д) область применения результатов;

е) рекомендации по внедрению или итоги внедрения результатов;

ж) экономическую эффективность или значимость работы;

з) прогнозные предположения о развитии объекта исследования.

Для выделения структурных частей реферата используют абзацные отступы. Если отчет не содержит сведений по какой-либо из перечисленных частей, то в тексте реферата она опускается, при этом последовательность изложения сохраняется. Оптимальный объем текста реферата — 850 печатных знаков, но не более одной страницы.

Для оформления реферата настроим окружение abstract:

```
\renewenvironment{abstract}{
   \clearpage
   {\ElementFormat\abstractname\par} }{}
```

Оно сформирует новую страницу и напечатает заголовок «РЕФЕРАТ». Определение декларации `\ElementFormat`, печатающей заголовок полужирным шрифтом посредине строки, приведено далее. Текст реферата верстается обычным образом, пример его оформления показан на рис. Г.3.

РЕФЕРАТ

Отчет 85 с., 1 кн., 24 рис., 12 табл., 50 источн., 2 прил.
РАСХОДОМЕРНЫЕ УСТАНОВКИ, ПОРШНЕВЫЕ РАСХОДОМЕРЫ, ТАХОМЕТРИЧЕСКИЕ РАСХОДОМЕРЫ, ИЗМЕРЕНИЕ, БОЛЬШИЕ РАСХОДЫ, ГАЗЫ

Объектом исследования являются поршневые установки для точного воспроизведения и измерения больших расходов газа.

Цель работы — разработка методики метрологических исследований установок и нестандартной аппаратуры для их осуществления.

В процессе работы проводились экспериментальные исследования отдельных составляющих и общей погрешности установок…

Рис. Г.3. Пример реферата отчета о НИР

Содержание

Содержание дожно включать введение, наименование всех разделов и подразделов, пунктов (если они имеют наименование), заключение, список использованных источников и наименования приложений. Каждую запись содержания оформляют как отдельный абзац, выровненный влево. После заголовков элементов ставят отточие и приводят номера страниц, на которых начинаются элементы. Номера страниц указывают выровненными по правому краю поля. Обозначения подразделов и пунктов приводят после абзацного отступа, равного соответственно

двум и четырем знакам, относительно обозначения разделов. Продолжение заголовков на второй и последующей строках выполняют, начиная от уровня начала этого заголовка на первой строке.

Содержание отчета объемом не более 10 с. можно не составлять. В отчете, состоящем из двух и более книг, каждая должна иметь свое содержание. При этом в первой книге помещают содержание всего отчета с указанием номеров книг, а в последующих — только содержание соответствующей книги. В первой книге можно указать только наименования последующих книг, опустив их содержание.

Верстку содержания нужно настроить. Класс report печатает заголовок «Оглавление» крупным полужирным шрифтом слева, ГОСТ же регламентирует центрированный заголовок «СОДЕРЖАНИЕ», напечатанный полужирным шрифтом основного размера. Заменим заголовок оглавления и заодно отредактируем остальные заголовки:

```
\AtBeginDocument{
   \renewcommand\abstractname{РЕФЕРАТ}
   \renewcommand\contentsname{СОДЕРЖАНИЕ}
   \renewcommand\listfigurename{СПИСОК ИЛЛЮСТРАЦИЙ}
   \renewcommand\listtablename{СПИСОК ТАБЛИЦ}
   \renewcommand\bibname{СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ}
   \renewcommand\figurename{Рисунок}
   \renewcommand\tablename{Таблица}
   \renewcommand\chaptername{Раздел}
   \thispagestyle{empty} }
```

Напомним, что пакет babel настраивает стандартные заголовки при входе в окружение document, поэтому наши настройки помещены в команду `\AtBeginDocument`. В конце списка добавлена команда `\thispagestyle{empty}`, автоматически отменяющая нумерацию первой (титульной) страницы, поэтому в рукописи ее можно опустить.

Отбивки элементов оглавления классом report не соответствуют требованиям ГОСТа. Изменим их средствами, предназначенными для этой цели в ядре LaTeX. Элемент оглавления печатает команда

`\@dottedtocline{`*уровень*`}{`*ширина отступа*`}`
`{`*ширина номера*`}{`*заголовок*`}{`*номер страницы*`}` .

В ее первом аргументе указывают код уровня раздела (см. табл. 1.2), во втором — ширину отступа элемента от края страницы, в третьем — ширину поля, отводимого под номер элемента оглавления. В четвертый аргумент помещают заголовок раздела, а в пятый — номер страницы, на которой он начинается. Для печати элементов глав, разделов и подразделов эту команду настраивают команды `\l@chapter` , `\l@section` и `\l@subsection` . Вот как они определяются в нашем случае:

```
\def\l@chapter{\@dottedtocline{0}{0pt}{1.5em}}
\def\l@section{\@dottedtocline{1}{1.25em}{2em}}
\def\l@subsection{\@dottedtocline{2}{3.25em}{2.75em}}
```

Конкретные значения заголовков и номеров страниц формируют обсуждавшиеся ранее команды `\addcontentsline`.

Термины и определения. Перечень сокращений и обозначений

Структурный элемент «ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ» содержит определения, необходимые для уточнения или установления используемых терминов. Их перечень начинают со слов: «В настоящем отчете о НИР применяют следующие термины с соответствующими определениями». Перечень оформляют в виде списка терминологических статей, располагаемых столбцом без знаков препинания в конце. Слева без абзацного отступа в алфавитном порядке приводят термины, справа через тире — их определения. Допустимо оформление перечня в виде таблицы из двух колонок: термин, определение.

Если в отчете используется более трех условных обозначений, требующих пояснения, составляют их перечень, при их количестве менее трех, необходимые сведения указывают в тексте отчета или в подстрочном примечании при первом упоминании. Структурный элемент «ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ» начинают со слов: «В настоящем отчете о НИР применяют следующие сокращения и обозначения». Перечень должен располагаться столбцом без знаков препинания в конце строки. Слева без абзацного отступа в алфавитном порядке приводят сокращения, условные обозначения, символы, единицы физических величин, а справа через тире — их детальную расшифровку.

Допускается свести определения, обозначения и сокращения в один структурный элемент «ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ».

Для оформления объединенного перечня воспользуемся средствами формирования списка аббревиатур, описанными в разд. 11.3 и прил. А.7. Так как отчет не содержит указателя, сверстаем перечень вместо него. Для этого загрузим пакет xindex со следующими настройками:

```
\usepackage[imakeidx]{xindex}
\makeindex[options= -n -g -a -c abbreviations]
```

В рукописи элементы перечня следует поместить в команды `\index{...}`, тогда команда `\printindex` будет автоматически генерировать его:

```
\chapter*{ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ}
В настоящем отчете о НИР применяют следующие...
%\input{\jobname.ind}
    ...
\printindex
```

Загружать перечень следует командой `\input`, разместив ее после команды `\chapter*`, которая напечатает заголовок и внесет перечень в оглавление. Перед первой загрузкой нужно создать перечень командой `\printindex`, которую нужно поставить в конце рукописи перед командой `\end{document}`, затем закомментировать ее и раскомментировать команду `\input`. Для обновление перечня достаточно будет раскомментировать команду `\printindex`, скомпилировать документ и вновь закомментировать ее.

Не забудьте поместить в папку с отчетом или в локальную папку с ресурсами xindex файл xindex-abbreviations.cfg, описанный в прил. А.7.

Введение

Введение должно содержать оценку современного состояния решаемой научно-технической проблемы, основание и исходные данные для разработки темы, обоснование необходимости проведения НИР, сведения о планируемом научно-техническом уровне разработки и патентных исследованиях, а также выводы из них, сведения о метрологическом обеспечении НИР. Во введении должны быть отражены актуальность и новизна темы, связь данной работы с другими НИР. Для промежуточного отчета должны быть указаны цели и задачи исследований, выполненных на данном этапе, их место в выполнении НИР в целом. В заключительном отчете приводят перечень наименований всех подготовленных промежуточных отчетов по этапам и их регистрационные номера, если они были представлены для регистрации в соответствующий орган (ЦИТиС, ЕГИСУ НИОКТР).

Основная часть и структурные элементы отчета

Наименования «СПИСОК ИСПОЛНИТЕЛЕЙ», «РЕФЕРАТ», «СОДЕРЖАНИЕ», «ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ», «ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ», «ВВЕДЕНИЕ», «ЗАКЛЮЧЕНИЕ», «СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ», «ПРИЛОЖЕНИЕ» служат заголовками структурных элементов. Их следует печатать в середине строки без точки в конце, прописными буквами, не подчеркивая. Каждый структурный элемент и раздел отчета начинают с новой страницы.

Основная часть отчета должна содержать:

– выбор направления исследований, включающий обоснование направления исследования, методы решения задач и их сравнительную оценку, описание выбранной общей методики проведения НИР;

– процесс теоретических и (или) экспериментальных исследований, включая определение характера и содержания теоретических исследований, методы исследований и расчетов, обоснование необходимости проведения экспериментальных работ, принципы действия разработанных объектов, их характеристики;

– обобщение и оценку результатов исследований, включая оценку полноты решения поставленной задачи и предложения по дальнейшим направлениям работ, оценку достоверности полученных результатов и технико-экономической эффективности их внедрения и их сравнение с аналогичными результатами отечественных и зарубежных работ, обоснование необходимости проведения дополнительных исследований, отрицательные результаты, приводящие к необходимости прекращения дальнейших исследований.

Единицы физических величин приводят по ГОСТ 8.417-2002 [44].

Основную часть отчета разбивают на разделы, подразделы и пункты и при необходимости подпункты. Разделы нумеруют в пределах всего отчета арабскими цифрами, а подразделы, пункты и подпункты, соответственно, в пределах раздела, подраздела и пункта. Номера составляют из номеров разделов, подразделов пунктов и подпунктов, разделенных точками, и печатают с абзацного отступа. Разделы и подразделы *должны*, а пункты *могут* иметь заголовки, которые печатают за номерами с прописной буквы, полужирным шрифтом, не подчеркивая, без переносов и точки в конце. Между номерами и заголовками точка не ставится. Если заголовок включает несколько предложений, их разделяют точками.

Класс report в основном обеспечивает требования, предъявляемые к нумерации разделов и составлению оглавлений, нужно лишь увеличить глубину нумерации командой `\setcounter{secnumdepth}{4}` (см. разд. 7.4). Более серьезные настройки требует печать заголовков. Стандартные классы LaTeX верстают их без отступа в виде таблицы из двух колонок, выводя в первую номер, а во вторую — заголовок, ГОСТ же требует, чтобы они формировались как обычные абзацы. Упоминавшийся ранее пакет titlesec не поддерживает такой стиль заголовков, поэтому нам придется настроить их, используя низкоуровневые команды ядра LaTeX.

Определим ряд команд, чтобы использовать в дальнейшем и прежде всего создадим длину, в которую сохраним ширину отступа. Дело в том,

что некоторые конструкции LaTeX зануляют длину \parindent, и нередко ее сохранение в независимой команде дает возможность восстановить отступ там, где он подавлен, например в командах разделов.

```
\newlength\ParIndent
\setlength\parindent{1.25cm}
\ParIndent=\parindent
\def\TitleFont{\normalfont \normalsize \bfseries \boldmath}
\def\SectionFormat{\hspace*{\ParIndent}\TitleFont\raggedright}
\def\ChapterFormat{\indent\TitleFont\raggedright}
\def\ElementFormat{\TitleFont\centering}
\def\@seccntformat#1{\csname the#1\endcsname\hspace{0.5em}}
```

Команда \ParIndent будет хранить ширину отступа. Команда \TitleFont будет устанавливать для печати текста и формул в заголовках разделов полужирный шрифт основного размера. Команды \SectionFormat, \ChapterFormat и\ElementFormat будут настраивать печать заголовков. Находящиеся в них декларации \raggedright и \centering не только выровняют их, но и подавят переносы слов. В конце мы отредактировали команду, печатающую номера разделов, `\@seccntformat{`*наименование раздела*`}`. По умолчанию она отбивает их справа пробелом шириной 1em, который в требуемом стиле оформления заголовков выглядит слишком широким, поэтому мы уменьшили его до 0.5em. *Наименование*, подставляемое в команду \@seccntformat, является текстом, который конструкция `\csname текст \endcsname` превращает в команду.[2] С ее помощью \@seccntformat формирует команды, выводящие значения счетчиков, например \thesubsection для счетчика подразделов.

Разделы формирует низкоуровневая команда LaTeX

```
\@startsection {наименование}{уровень}{ширина отступа}
               {верхняя отбивка}{нижняя отбивка}{печать}
               *[краткий заголовок]{заголовок}.
```

Ее первым аргументом является упоминавшееся выше наименование раздела, а вторым — его уровень, задаваемый числом от минус единицы до пяти (см. табл. 1.2). В трех следующих аргументах указывают ширину отступа и высоту верхней и нижней отбивки заголовка. Длины могут быть как положительны, так и отрицательны. При отрицательной ширине отступа заголовок целиком смещается к левому краю страницы. Если отрицательна верхняя отбивка, первый абзац раздела верстается без отступа, а заголовок отбивается на указанную *положительную* величину. Если отрицательна нижняя отбивка, заголовок не выносится в отдельную строку, а располагается в

[2] Указанная конструкция делает из текста имя команды и, если такая команда существует, она выполняется, в противном случае выдается сообщение, что команда не определена.

начале первого абзаца раздела. В этом случае длина (опять же *положительная*) задает ширину пробела между ним и текстом абзаца. Шестой аргумент настраивает печать заголовка. Как правило, в нем указывают параметры шрифта и метод выравнивания. В последний аргумент помещают заголовок, а в параметр — его вариант для колонтитулов и оглавления. Если перечень аргументов содержит звездочку, раздел не нумеруется и не заносится в оглавление. Звездочка и параметр могут отсутствовать в вызове команд `\@startsection`.

Приступим к настройке команд:

```
\def\section{\@startsection{section}{1}{-\ParIndent}
    {\bigskipamount}{\medskipamount}{\SectionFormat}},
\def\subsection{\@startsection{subsection}{2}{-\ParIndent}
    {\bigskipamount}{\medskipamount}{\SectionFormat}},
\def\subsubsection{\@startsection{subsubsection}{3}{0pt}
    {\medskipamount}{-0pt}{\indent\TitleFont}{}}.
```

Аргумент `-\ParIndent` команды `\@startsection` прижимает заголовки команд `\section` и `\subsection` к левой границе страницы, а пробел `\hspace*`, находящийся в команде `\SectionFormat`, формирует начальный отступ. Перед заголовками, верстаемыми с левым выравниваем шрифтом `\TitleFont`, вставляется отбивка высотой `\bigskipamount`, а после них — высотой `\medskipamount`. Аргумент вместе с возможным параметром и звездочкой, «недостающие» у команды `\@startsection`, наследуются у команд `\section` и `\subsection`. Команда `\subsubsection` сформирована таким образом, что заголовок подпункта создается непосредственно в начале абзаца. В данном случае мы полностью определили команду `\@startsection`, задав ей пустой последний аргумент, чтобы команда `\subsubsection` печатала только номер подпункта, отделенный от последующего текста пробелом шириной 0.5em. От предыдущего текста подпункт отделяет пробел высотой `\medskipamount`. Выполненные настойки изменяют лишь формат печати заголовков подразделов, пунктов и подпунктов отчета, полностью сохраняя синтаксис создающих их команд, описанный в разд. 1.5.4.

Команду `\chapter` определим так, чтобы она создавала разделы отчета и его элементы. Для этого сначала введем команду, которая будет верстать заголовки, передаваемые ей в качестве аргументов:

```
\def\ChapterTitle#1#2{
   \addcontentsline{toc}{chapter}
      {\protect\numberline{\thechapter} #1}
   {\ChapterFormat\thechapter\hspace{.5em} #2\par}
   \vskip\medskipamount \@afterheading }
```

Команда `\addcontentsline{•}{•}{•}` заносит раздел в оглавление (см. разд. 8.5). Ее последний аргумент перенесен на следующую строку вместе с открывающей скобкой. Вплоть до вывода аргумента `#1` он не содержит

пробелов, так как пробелы, находящиеся перед командой \numberline или перед номером раздела, выводимом командой \thechapter, сдвинут строки оглавления от края страницы.

Декларация \ChapterFormat и команда \par печатают заголовок в начале раздела. Действие декларации ограничено группирующими скобками, чтобы ее настройки не оказали влияние на последующий текст. Команда \par не только оформляет заголовки в виде абзацев, но и инициирует действие декларации, задающей их выравнивание.

Последние команды отбивают заголовки снизу, следя за тем, чтобы высота отбивки не оказалась избыточной. Команда \vskip{*длина*} аналогично команде \vspace вставляет вертикальный пробел, но в отличие от нее она взаимодействует с еще одной командой \addvspace{*длина*}, также вставляющей пробел. Когда команда \addvspace следует за \vskip, вставляется лишь один из пробелов, имеющий максимальную *длину*. Если вслед за разделом создается подраздел, вернюю отбивку которого задает команда \addvspace, команда \@afterheading «связывает» ее с командой \vskip. Если же за заголовком раздела следует текст, вставляется пробел \vskip.

Специальная конструкция определяет команды \part[•]{•}, \part*{•}, \chapter[•]{•}, \chapter*{•}, формирующие главы и части документов:

\def\chapter{ \clearpage \secdef *команда1 команда2* }.

Команда \clearpage завершает предыдущую страницу, выводя все плавающие объекты, чтобы рисунки и таблицы предыдущей главы не попали в следующую. Команда \secdef анализирует имя вызывающей команды и, в зависимости от того, оканчивается оно звездочкой или нет, выполняет одну из *команд*. *Команда1* может иметь аргумент и параметр, а *команда2*, соответствующая «команде со звездочкой», должна иметь только аргумент. Воспользуемся этой конструкцией в нашем определении:

```
\def\chapter{ \clearpage \secdef \@chapter \@element }
\def\@element#1{
    \phantomsection \addcontentsline{toc}{chapter}{#1}
    {\ElementFormat #1 \par}
    \vskip\medskipamount \@afterheading }
\def\@chapter[#1]#2{
    \refstepcounter{chapter} \ChapterTitle{#1}{#2}
    \typeout{\chaptername\space\thechapter.}
    \addtocontents{lof}{\medskip}
    \addtocontents{lot}{\medskip} }
```

Команды \chapter* будут создавать элементы отчета, инициируя выполнение команды \@element. Она внесет их в оглавление на уровне глав, выполнив перед этим команду \phantomsection, которая обеспечит корректность гиперссылок в электронных документах (см. разд. 7.8), а затем подобно

команде \ChapterTitle напечатает заголовки и отобьет их снизу. Отметим, что команды \chapter* используются в оформлении списков рисунков, таблиц и литературы, поэтому они автоматически попадут в оглавление.

Посредством команды \@chapter команды \chapter будут создавать разделы (главы). Перед версткой заголовков команда \refstepcounter обновит счетчик глав, а после их печати будет выполняться пара дополнительных действий. Команда \typeout{текст} выдаст на консоль и запишет в log-файл сообщение «Раздел …»,[3] а команды \addtocontents добавят небольшие вертикальные отбивки в списки lof и lot, разделяющие зрительно рисунки и таблицы разных разделов.

Описанные настройки формируют следующий стиль верстки разделов:

1 Заголовок раздела

1.1 Пример, демонстрирующий печать заголовка подраздела, занимающего несколько строк

1.1.1 Заголовок пункта

1.1.1.1 Подпункт не заносится в оглавление, а его текст печатается непосредственно за номером. Если понадобится поместить номер на отдельной строке, начните текст с нового абзаца.

Заключение

Заключение должно содержать:

– краткие выводы по результатам выполненной НИР или ее этапов;

– оценку полноты решений поставленных задач;

– разработку рекомендаций и исходных данных по конкретному использованию результатов НИР;

– результаты оценки технико-экономической эффективности внедрения;

– результаты оценки научно-технического уровня выполненной НИР в сравнении с лучшими достижениями в этой области.

Список использованных источников

Список использованных источников должен содержать сведения об источниках, использованных при составлении отчета, оформленных в соответствии с требованиями ГОСТ Р 7.0.100-2018 и ГОСТ Р 7.0.80-2023 [100, 101]. Сведения следует располагать в порядке появления ссылок на них в тексте отчета, нумеровать арабскими цифрами

[3] Стандартные классы сообщают о формировании новых частей и глав.

с точкой и печатать с абзацного отступа. Ссылки оформляют арабскими цифрами в квадратных скобках. Сведения об интернет-ресурсах должны содержать дату последнего обращения (см. с. 148).

Регламентированный стиль цитирования соответствует стандартным настройкам пакета cite, который следует загрузить при верстке отчета. Для генерации списка литературы подходит стиль ugost2008 пакета gost, однако формат его печати окружением thebibliography не соответствует требованиям ГОСТа. В стандартных настройках окружение заключает номера источников в скобки, выделяя их в отдельное левое поле, а сами источники разделяет небольшим вертикальным пробелом, тогда как согласно ГОСТ 7.32-2017 номера должны заканчиваться точкой, а описания источников представляют собой обычные абзацы с отступом. Замена скобок точкой обсуждалась на с. 148, осталось настроить отбивки. Для этого отредактируем окружение thebibliography:

```
\let\SavedBibliography\thebibliography
\renewenvironment{thebibliography}[1]{
   \begin{SavedBibliography}{\hspace{-0.8em}}
      \itemsep=0pt
      \parskip=0pt
      \itemindent=\ParIndent }
   { \end{SavedBibliography} }
```

Команда `\let` «копирует» его в окружение SavedBibliography, после чего декларация `\renewenvironment` изменяет его (см. разд. 7.2) следующим образом. Сохраненное окружение SavedBibliography устанавливает стандартные параметры верстки библиографии (см. разд. 8.10), а команды, находящиеся внутри окружения, их корректируют. Для формирования отступов в списке литературы длина `\itemindent` приравнивается длине `\ParIndent`. Зануление длин `\itemsep` и `\parskip` убирает ненужную вертикальную отбивку.

Для полной совместимости с обычным окружением thebibliography редактируемое тоже имеет аргумент, однако он фиктивен, так как не передается окружению SavedBibliography. В его аргумент помещен отрицательный пробел, корректирующий левую отбивку списка литературы.[4]

Отметим важную деталь. Если отчет верстается в классе extreport, нужно обязательно загрузить пакет hyperref, иначе шрифт адресов интернет-ресурсов в библиографии будет слишком мелок.

[4] Ширину отбивки задает число источников в списке литературы, указываемое в аргументе окружения thebibliography, а также элементы, окружающие номера источников. В нашем случае элементом служит точка, проставляемая после номеров, ширину которой компенсирует отрицательный пробел.

Приложения

Приложения могут включать:

– дополнительные материалы к отчету;

– промежуточные математические доказательства и расчеты;

– таблицы вспомогательных цифровых данных;

– протоколы испытаний;

– заключение метрологической экспертизы;

– инструкции, методики, описания алгоритмов и программ, разработанных в процессе выполнения НИР;

– иллюстрации вспомогательного характера;

– копии технического задания на НИР, программы работ или другие исходные документы для выполнения НИР;

– протокол рассмотрения результатов выполненной НИР на научно-техническом совете;

– акты внедрения результатов НИР или их копии;

– копии охранных документов.

Приложения оформляют как продолжение отчета на последующих его листах или в виде отдельной книги. Если в составе НИР предусмотрено проведение патентных исследований, приложения могут быть включены в отчет о патентных исследованиях. Приложения должны иметь общую с остальной частью отчета сквозную нумерацию страниц. Все приложения должны быть перечислены в содержании с указанием их обозначений, статуса и наименования.

Приложения располагают в порядке ссылок на них в тексте отчета, который должен содержать ссылки на все приложения. Приложения начинают с новой страницы, вверху которой по центру отдельной строкой без точки в конце с прописной буквы, полужирным шрифтом записывают заголовок «ПРИЛОЖЕНИЕ» и его обозначение прописными буквами кириллического алфавита, начиная с А, за исключением букв Ё, З, Й, О, Ч, Ъ, Ы, Ь. Допускается обозначать приложения буквами латинского алфавита, за исключением букв I и O, а в случае полного использования букв кириллического или латинского алфавита арабскими цифрами. Если в отчете одно приложение, оно обозначается «ПРИЛОЖЕНИЕ А».

Приложения, как правило, выполняют на листах формата А4, но также допускается оформление на листах формата А3, которые учитывают как одну страницу. Текст приложений можно разделить на разделы, подразделы, пункты, подпункты, которые нумеруют в пределах каж-

дого приложения. Перед номером ставится обозначение приложения, например «А.1».

Напомним, что создание приложений инициирует декларация `\appendix`. Настроим ее так, чтобы команды `\chapter` формировали их в соответствии с требованиям ГОСТа.

```
\\edef\appendix{ \unexpanded \expandafter {\appendix
   \def\chaptername{Приложение}
   \def\thechapter{\Asbuk{chapter}}
   \def\ChapterTitle#1#2{
      \addcontentsline{toc}{chapter}
         {\protect\numberline{#1\hspace{.5em}\thechapter}
            \hphantom{#1}}
      {\ElementFormat#2\space\thechapter\par}
      \vskip\medskipamount \@afterheading } } }
```

Не вдаваясь в детали, скажем, что конструкция, находящаяся в первой строке, переопределит декларацию `\appendix` таким образом, что сначала выполняется она сама, а затем команды, добавленные в ее определение. Первая из них установит «Приложение» в качестве заголовка сообщений, выводимых на консоль и в log-файл, вторая установит заглавные русские буквы для формата печати счетчика (см. с. 171), третья изменит формат вывода заголовков. Команда `\ChapterTitle` будет печатать заголовки *перед* номерами. Вследствие этого в оглавлении номера вылезут за пределы отведенного для них поля,[5] и на них наложится отточие. Чтобы это не произошло, команда `\hphantom` сдвигает отточие на ширину заголовков.

Настроенные таким образом команды `\chapter{ПРИЛОЖЕНИЕ}` будут создавать «**ПРИЛОЖЕНИЕ А**», «**ПРИЛОЖЕНИЕ Б**» и т. д.

Иллюстрации и таблицы

Иллюстрации и таблицы следует располагать непосредственно после текста, где они упоминаются впервые, или на следующей странице. На все иллюстрации и таблицы в отчете должны быть даны ссылки, в которых следует печатать слово «рисунок» или «таблица» и номер. Иллюстрации и таблицы следует нумеровать арабскими цифрами сквозной нумерацией или в пределах раздела отчета. В последнем случае номер составляют из номера раздела и порядкового номера иллюстрации, разделенных точкой. Иллюстрации и таблицы каждого приложения нумеруют отдельно и обозначают арабскими цифрами с добавлением перед цифрой обозначения приложения: Рисунок А.3.

[5] Определенная ранее команда `\l@chapter` устанавливает ширину поля, исходя из нумерации разделов, не превышающей двух цифр.

Чертежи, графики, диаграммы, схемы, помещаемые в отчете, должны соответствовать требованиям стандартов Единой системы конструкторской документации (ГОСТ Р 2.105-2019 [43]). Иллюстрации могут иметь наименование (подпись) и пояснительные данные (подрисуночный текст). Слово «Рисунок», его номер и через тире наименование помещают *после* пояснительных данных и располагают в центре под рисунком без точки в конце, например:

Рисунок 2 – Оформление таблицы

Номера и подписи таблиц следует помещать слева над ними. Подписи, состоящие из нескольких строк, записывают через один межстрочный интервал с прописной буквы без точки в конце. Перенос слов в подписях не допускается.

Заголовки колонок и строк таблиц следует печатать с прописной буквы, а подзаголовки колонок — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков точку не ставят, а их названия указывают в единственном числе. Заголовки колонок выравнивают по центру, а заголовки строк — по левому краю. Не допускается разделять заголовки и подзаголовки боковика[6] и колонок диагональными линиями.

Таблицы слева, справа, сверху и снизу ограничивают линиями, при этом горизонтальные и вертикальные линии допускается не проводить, если их отсутствие не затрудняет пользование таблицей.

Текст, повторяющийся в строках одной и той же колонки и состоящий из одиночных слов, при первом повторении заменяют словами «то же», а далее кавычками. Не допускается ставить кавычки вместо повторяющихся цифр, буквенно-цифровых обозначений, знаков и символов.

Длинную таблицу можно переносить на другую страницу, при этом слово «Таблица», ее номер и наименование указывают один раз слева над первой частью таблицы, а над другими частями также слева пишут слова «Продолжение таблицы» и указывают ее номер. При делении таблицы на части ее головку или боковик можно заменить, соответственно, номерами колонок и строк, при этом арабскими цифрами нумеруют колонки и (или) строки и первой части таблицы.

В таблицах допускается применять размер шрифта меньше, чем в тексте отчета.

[6] Боковик — левая колонка таблицы, содержащая заголовки строк.

Чтобы нумеровать рисунки, таблицы и формулы по разделам, свяжем их номера с номерами разделов декларациями

```
\counterwithin{figure}{chapter},
\counterwithin{table}{chapter},
\counterwithin{equation}{chapter}.
```

Печать подписей таблиц и рисунков настроим средствами пакета caption [42]. Не обсуждая его богатые возможности, приведем лишь основную декларацию: `\captionsetup[`*тип подписи*`]{`*настройки*`}` . В параметре указывают тип настраиваемой подписи: рисунок или таблица. Декларации, у которых отсутствует параметр, задают общие настройки.

Требованиям ГОСТа удовлетворяют следующие настройки:

```
\usepackage{caption}
\captionsetup{font=normalsize,labelsep=endash}
\captionsetup[figure]{justification=centering}
\captionsetup[table]{singlelinecheck=off,justification=raggedright},
```

Первая декларация установит для подписей основной размер шрифта и инициирует использование короткого тире в качестве разделителя номеров и заголовков. Вторая декларация отцентрирует подписи рисунков и отключит перенос слов.[7] Третья декларация обеспечит левое выравнивание подписей таблиц и также отключит перенос. В таких настройках подписи таблиц будут выровнены по левому краю *страницы*, а чтобы привязать их к границе *таблицы*, поместите ее вместе с подписью в министраницу.

Если отчет содержит большое количество иллюстраций и таблиц, имеет смысл привести в отчете их списки, хотя ГОСТ этого и не требует. Их можно разместить после библиографии или даже в самом конце после приложений, предварительно объединив следующим образом:

```
\edef\listoffigures{ \unexpanded \expandafter {\listoffigures
                  { \let\clearpage\relax \bigskip \listoftables} } }
```

Данный набор команд отменит вывод списка таблиц на отдельную страницу и поместит его в конце списка иллюстраций, отделив пробелом `\bigskip`. В результате команда `\listoffigures` станет печатать оба списка.

Формулы

В наборе формул следует учитывать, что длинные уравнения разбивают на строки после операторов (+, −, :, × и т. д.) или знаков сравнения (=, <, > и т. п.). На новой строке знак повторяют. При разбиении на знаке умножения, применяют символ «×».

[7] Напомним, что декларации `\centering`, `\raggedright` и `\raggedleft` отключают переносы.

Пояснение значений символов и числовых коэффициентов приводят непосредственно под формулой в той же последовательности, в которой они представлены в формуле. Первую строку пояснения начинают со слова «где» без двоеточия с абзаца. Значение каждого символа и числового коэффициента приводят с новой строки.

В приложениях формулы нумеруются арабскими цифрами в пределах приложения с добавлением его обозначения, например (В.1). В основном тексте допускается нумерация в пределах раздела.

Формулы особой настройки не требуют. Нужно лишь позволить их разбиение между страницами декларацией `\allowdisplaybreaks`. Как привязать нумерацию формул к разделам, показано ранее.

Перечисления

Приводимые в отчете перечисления представляют собой обычные абзацы. Перед каждым их элементом следует ставить тире. Чтобы ссылаться на элементы, вместо тире используют строчные буквы русского алфавита, начиная с буквы «а» (за исключением букв ё, з, й, о, ч, ъ, ы, ь), или арабские цифры, после которых ставят скобку. Простые перечисления отделяют запятой, сложные — точкой с запятой. Вложенные перечисления дополнительно не отбиваются, в их обозначениях используются другие элементы (буквы или цифры).

Перечисления данного раздела для примера оформлены в соответствии с ГОСТом.

Примечания и сноски

Примечания приводят, если необходимы пояснения или справочные данные к содержанию текста, таблиц или графического материала. Слово «Примечание» следует печатать с прописной буквы с абзацного отступа, не подчеркивая. Примечания следует помещать непосредственно после материалов, к которым они относятся. Если примечание одно, то после слова «Примечание» ставят тире и текст, печатаемый с прописной буквы. Такое примечание не нумеруется. Несколько примечаний оформляют в виде пречисления, пронумерованного арабскими цифрами без скобки. Перед ним печатают слова «Примечания», а в конце его элементов ставят точки.

Примечание можно оформить в виде сноски. Номер сноски указывают надстрочно арабскими цифрами без пробела непосредственно после того слова, числа, символа, предложения, к которому дается пояснение.

Вместо цифр допускается использовать звездочку *. Сноску начинают с абзацного отступа, располагают в конце страницы, на которой приведен поясняемый объект, и отделяют от текста короткой сплошной тонкой горизонтальной линией, проводимой с левой стороны страницы.

Для оформления сносок можно пользоваться стандартной командой `\footnote`, но ГОСТ не предусматривает уменьшение их шрифта, поэтому в преамбуле нужно переопределить команду `\@makefntext{`*текст*`}`, печатающую *текст* сносок:

```
\def\@makefntext#1{\normalsize \vspace{.5ex} \hspace{\ParIndent}
                                  \textsuperscript{\small\@thefnmark} #1 }
```

Сначала она установит нужный размер шрифта, затем сделает небольшую вертикальную отбивку и отступ, после чего выведет номер сноски и ее текст. Межстрочный интервал сносок жестко не регламентирован. По умолчанию они будут печататься через один интервал, а если понадобится, его можно увеличить, добавив в команду декларацию `\onehalfspacing`.

Г.2. Верстка отчета

Отчет верстается как обычный документ класса report шрифтом 14pt через полтора интервала. Библиография верстается в стиле ugost2008.

Настройки выполнены так, чтобы автор мог оперировать только стандартными командами, синтаксис которых не изменен. Это позволит копировать текст в другие документы, не опасаясь появления ошибок.

Список исполнителей можно сделать с помощью окружений tabular или longtable. Для верстки реферата настроено окружение abstract. Примеры компоновки титульного листа, списка исполнителей и реферата показаны на рис. Г.1, Г.2 и Г.3.

Старшим уровнем разделов являются главы, создаваемые командами `\chapter`. Их можно разбить на три уровня командами `\section`, `\subsection` и `\subsubsection`. Приложения следует формировать командами `\chapter{ПРИЛОЖЕНИЕ}`, а элементы отчета — командами `\chapter*{`•`}`. Элементы отчета автоматически вносятся в оглавление.

Пакеты xindex и imakeidx помогают автоматически сформировать перечень определений, обозначений и сокращений. Для этого нужно при упоминании в рукописи задать их расшифровки в командах `\index{...}`. Файл с перечнем будет наследовать имя файла рукописи отчета. Генерация и загрузка подобного файла обсуждалась в разд. 11.3.

Не забудьте поместить в папку с отчетом или в локальную папку с ресурсами xindex файл xindex-abbreviations.cfg, описанный в прил. А.7.

Приведенный далее макет содержит пакеты, настроенные для верстки отчета. В зависимости от его тематики список пакетов можно расширить, добавив ресурсы, требуемые для описания выполненных работ.

Макет отчета

```
\documentclass[14pt,a4paper]{extreport}

\usepackage{geometry}
\geometry{left=30mm, right=15mm, top=20mm, bottom=20mm}

\usepackage{setspace}
\onehalfspacing

\usepackage[russian]{babel}
\usepackage{indentfirst}

\usepackage{caption}
\captionsetup{font=normalsize, labelsep=endash}
\captionsetup[table]{singlelinecheck=off, justification=raggedright}
\captionsetup[figure]{justification=centering}

\usepackage{graphicx}
\usepackage{array}
\usepackage{longtable}
\usepackage{booktabs}
\usepackage{cite}
\usepackage[hidelinks]{hyperref}

\usepackage[imakeidx]{xindex}
\makeindex[options= -n -g -a -c abbreviations]

\usepackage{amssymb}
\usepackage{mathtools}
\allowdisplaybreaks

\usepackage[mathbf=sym, mathsf=sym]{unicode-math}
\setmainfont{TimesNewRoman}
\setsansfont{Calibri}
\setmonofont{CourierNew}
\setmathfont{XITSMath}
\setmathfont{XITSMath}[range={it/greek->up,bfit/greek->bfup}]

\usepackage{gost-report}

\bibliographystyle{ugost2008}
```

```
\begin{document}
\begin{center}
\singlespacing
   ...   титульная страница   ...
\newpage
\textbf{СПИСОК ИСПОЛНИТЕЛЕЙ} \\[2ex]
\begin{tabular}{ >\raggedright b{.43\textwidth}
                 !{ \rule{.21\textwidth}{.4pt} }
                 >\raggedright p{.27\textwidth} }
   Руководитель НИР, \linebreak ... &
   И.О. Фамилия \linebreak\footnotesize (...) \tabularnewline
   ...   Исполнители и нормоконтролер   ...
\end{tabular}
\end{center}
\begin{abstract}
   ...   Реферат   ...
\end{abstract}
\tableofcontents     %%% Содержание
%\chapter*{ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ}
%В настоящем отчете о НИР применяют следующие сокращения,
%обозначения и термины с соответствующими определениями:
%\input{\jobname.ind}
\chapter*{ВВЕДЕНИЕ}
            ...
\chapter{...}
   ...   Основной текст отчета   ...   \index{...}   ...
\chapter*{ЗАКЛЮЧЕНИЕ}
            ...
\appendix
\chapter{ПРИЛОЖЕНИЕ}
\section{...}
   ...   Текст приложений   ...
\bibliography{...}   %%% Список литературы
\listoffgures        %%% Списки рисунков и таблиц
\printindex
\end{document}
```

В стилевом файле (пакете) gost–report.sty собраны настройки, необходимые для выполнения требований ГОСТ 7.32-2017. Он должен находиться в папке с рукописью или в папке с локальными ресурсами LaTeX (см. прил. А.3). Его синтаксис учитывает, что в пакетах символ @ имеет статус буквы, широко используемой в именах служебных команд, однако перед версткой ему присваивается статус обычных символов, которых не должно быть в именах команд «на уровне пользователя».

gost–report.sty

```
\AtBeginDocument{
   \renewcommand\contentsname{СОДЕРЖАНИЕ}
   \renewcommand\listfigurename{СПИСОК ИЛЛЮСТРАЦИЙ}
   \renewcommand\listtablename{СПИСОК ТАБЛИЦ}
   \renewcommand\bibname{СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ}
   \renewcommand\figurename{Рисунок}
   \renewcommand\tablename{Таблица}
   \renewcommand\chaptername{Раздел}
   \thispagestyle{empty} }

\counterwithin{figure}{chapter}
\counterwithin{table}{chapter}
\counterwithin{equation}{chapter}
\setcounter{secnumdepth}{4}

\newlength\ParIndent
\setlength\parindent{1.25cm}
\ParIndent=\parindent
\def\TitleFont{\normalfont \normalsize \bfseries \boldmath}
\def\SectionFormat{\hspace*{\ParIndent}\TitleFont\raggedright}
\def\ChapterFormat{\indent\TitleFont\raggedright}
\def\ElementFormat{\TitleFont\centering}

\def\l@chapter{\@dottedtocline{0}{0pt}{1.5em}}
\def\l@section{\@dottedtocline{1}{1.25em}{2em}}
\def\l@subsection{\@dottedtocline{2}{3.25em}{2.75em}}

\def\@seccntformat#1{\csname the#1\endcsname\hspace{0.5em}}

\def\section{\@startsection{section}{1}{-\ParIndent}
     {\bigskipamount}{\medskipamount}{\SectionFormat}}

\def\subsection{\@startsection{subsection}{2}{-\ParIndent}
    {\bigskipamount}{\medskipamount}{\SectionFormat}}
```

```
\def\subsubsection{\@startsection{subsubsection}{3}{0pt}
      {\medskipamount}{-0pt}{\indent\TitleFont}{}}

\def\ChapterTitle#1#2{
   \addcontentsline{toc}{chapter}
      {\protect\numberline{\thechapter} #1}
   {\ChapterFormat\thechapter\hspace{.5em} #2\par}
    \vskip\medskipamount \@afterheading }

\def\chapter{ \clearpage \secdef \@chapter \@element }

\def\@element#1{
   \phantomsection\addcontentsline{toc}{chapter}{#1}
   {\ElementFormat #1 \par}
   \vskip\medskipamount \@afterheading }

\def\@chapter[#1]#2{
   \refstepcounter{chapter}
   \ChapterTitle{#1}{#2}
   \typeout{\chaptername\space\thechapter.}
   \addtocontents{lof}{\medskip}
   \addtocontents{lot}{\medskip} }

\edef\appendix{ \unexpanded \expandafter {\appendix
   \def\chaptername{Приложение}
   \def\thechapter{\Asbuk{chapter}}
   \def\ChapterTitle#1#2{
      \addcontentsline{toc}{chapter}
         {\protect\numberline{#1\hspace{.5em}\thechapter}
            \hphantom{#1}}
      {\ElementFormat #2\space\thechapter\par}
      \vskip\medskipamount \@afterheading } } }

\def\@biblabel#1{#1.}
\let\SavedBibliography\thebibliography
\renewenvironment{thebibliography}[1]{
   \begin{SavedBibliography}{\hspace{-0.8em}}
      \itemsep=0pt
      \parskip=0pt
      \itemindent=\ParIndent }
   { \end{SavedBibliography} }

\renewenvironment{abstract}{ \clearpage
   {\ElementFormat\abstractname\par} }{}
```

```
\def\@makefntext#1{ \normalsize
   \vspace{.5ex}\hspace{\ParIndent}
   \textsuperscript{\small\@thefnmark}#1 }

\edef\listoffigures{ \unexpanded \expandafter {\listoffigures
      { \let\clearpage\relax \bigskip \listoftables} } }
```

Г.3. Диссертация и автореферат

ГОСТ Р 7.0.11-2011 [47] устанавливает требования к верстке диссертаций и авторефератов, которые во многом одинаковы с теми, что предъявляются к оформлению отчетов о НИР, поэтому мы не будем обсуждать их столь же подробно.

Диссертацию на соискание ученой степени доктора наук представляют в виде специально подготовленной рукописи, научного доклада или опубликованной монографии. Диссертацию на соискание ученой степени кандидата наук представляют в виде специально подготовленной рукописи или опубликованной монографии. Автореферат диссертации — документ, напечатанный типографским способом, в котором автор кратко излагает основное содержание диссертации. Автореферат оформляют на диссертацию, представленную в виде рукописи и изданной монографии.

Диссертация в виде рукописи должна иметь твердый переплет. Ее следует печатать на одной стороне листа белой бумаги одного сорта формата А4 через полтора интервала размером шрифта 12–14 пунктов. Страницы диссертации должны иметь следующие поля: левое — 25 мм, правое — 10 мм, верхнее — 20 мм, нижнее — 20 мм. Абзацный отступ должен быть одинаковым по всему тексту и равен пяти знакам (12.5–17 мм). Все страницы диссертации, включая иллюстрации и приложения, нумеруют по порядку без пропусков и повторений. Первой страницей считается титульный лист, на котором номер не ставят. Номера печатают посредине верхнего поля страниц.

Диссертация в виде рукописи имеет следующую структуру.

а) титульный лист;

б) оглавление;

в) текст диссертации:

1) введение,

2) основная часть,

3) заключение;

г) список сокращений и условных обозначений;

д) словарь терминов;

е) список литературы;

ж) список иллюстративного материала;

и) приложения.

Элементы, выделенные жирным шрифтом, являются обязательными, а остальные — дополнительными.

Титульный лист

На титульном листе приводят следующие сведения:

– наименование организации, где выполнена диссертация;

– статус диссертации — «на правах рукописи»;

– фамилию, имя, отчество диссертанта;

– название диссертации;

– шифр и наименование специальности (по номенклатуре специальностей научных работников);

– искомую степень и отрасль науки;

– фамилию, имя, отчество научного руководителя или консультанта, ученую степень и ученое звание;

– место и год написания диссертации.

К диссертации прилагают дополнительный титульный лист на русском языке, если работа написана на другом языке.

Текст диссретации

Введение к диссертации включает в себя следующие основные структурные элементы:

– актуальность темы исследования;

– степень ее разработанности;

– цели и задачи;

– научную новизну;

– теоретическую и практическую значимость работы;

– методологию и методы исследования;

– положения, выносимые на защиту;

– степень достоверности и апробацию результатов.

В заключении диссертации излагают итоги выполненного исследования, рекомендации, перспективы дальнейшей разработки темы.

Основной текст должен быть разделен на главы и параграфы или разделы и подразделы, которые нумеруют арабскими цифрами. Каждую главу (раздел) начинают с новой страницы. Заголовки располагают посередине страницы без точки на конце. Переносить слова в заголовке не допускается. Заголовки отделяют от текста сверху и снизу тремя интервалами.

Рисунки, таблицы, формулы

Иллюстрации и таблицы оформляют в соответствии с требованиями ГОСТ Р 2.105 и размещают под текстом, в котором впервые дана ссылка на них, или на следующей странице, а при необходимости — в приложении к диссертации. Допускается использование приложений нестандартного размера, которые в сложенном виде должны соответствовать формату А4. Иллюстрации и таблицы нумеруют раздельно арабскими цифрами сквозной нумерацией или в пределах главы. На все иллюстрации и таблицы должны быть приведены ссылки в тексте диссертации. В ссылках следует писать слово «Рисунок» или «Таблица» с указанием номера. Перечень таблиц указывают в списке иллюстративного материала.

Иллюстрации при необходимости могут иметь наименование и пояснительные данные (подрисуночный текст). Слово «Рисунок» и его наименование, отделенное тире, помещают после пояснительных данных.

Высота строк таблиц должна быть не менее 8 мм. Заголовки их граф и строк следует писать с прописной буквы, а подзаголовки — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков точки не ставят. Заголовки и подзаголовки граф указывают в единственном числе.

Таблицы слева, справа и снизу, как правило, ограничивают линиями. Допускается не ограничивать таблицы линиями слева и справа. Горизонтальные и вертикальные линии, разграничивающие строки таблицы, также допускается не проводить, если их отсутствие не затрудняет пользование таблицей. В первой части таблицы, продолжающейся на следующей странице, нижнюю ограничивающую линию рекомендуется не проводить. Головка должна быть отделена от остальной части таблицы двойной линией.

Подпись таблицы указывают один раз слева над ее первой частью, над другими частями пишут слова «Продолжение таблицы» с указанием ее номера, при этом надпись «Продолжение таблицы» можно не указывать.

Формулы оформляют в соответствии с требованиями ГОСТ Р 2.105 и нумеруют арабскими цифрами сквозной нумерацией или в пределах главы. Формулы, помещаемые в приложениях, должны нумероваться отдельной нумерацией арабскими цифрами в пределах каждого приложения с добавлением перед каждой цифрой обозначения приложения. Номер заключают в круглые скобки и записывают на уровне формулы справа. Формулы, следующие одна за другой и не разделенные текстом, разделяют запятой. Пояснения символов и числовых коэффициентов, если они не пояснены ранее в тексте, должны быть приведены непосредственно под формулой. Пояснения каждого символа следует давать с новой строки в той последовательности, в которой символы приведены в формуле. Первая строка пояснения должна начинаться со слова «где» без двоеточия после него.

Списки сокращений, условных обозначений и терминов

Перечень сокращений и условных обозначений следует располагать столбцом. Слева в алфавитном порядке или в порядке их первого упоминания в тексте приводят сокращения или условные обозначения, справа — их детальную расшифровку. Наличие перечня не исключает расшифровку сокращения и условного обозначения при первом упоминании в тексте. Перечень помещают после основного текста и указывают в оглавлении диссертации.

При использовании специфической терминологии в диссертации должен быть приведен список принятых терминов с соответствующими разъяснениями. Термин записывают со строчной буквы, а определение — с прописной. Термин отделяют от определения двоеточием. Список терминов помещают после перечня сокращений и условных обозначений и указывают в оглавлении диссертации.

Список литературы и цитирование

Список литературы должен включать библиографические записи, оформленные согласно ГОСТ Р 7.0.100. Он должен быть размещен после словаря терминов и указан в оглавлении диссертации. Допускаются

следующие способы группировки записей: алфавитный, систематический (тематический или в порядке первого упоминания в тексте) и хронологический. При алфавитном порядке группировки записи располагают по алфавиту фамилий авторов или первых слов заглавий документов. Библиографические записи произведений авторов-однофамильцев располагают в алфавите их инициалов. При наличии в списке литературы на других языках, кроме русского, образуется дополнительный алфавитный ряд, который располагают после изданий на русском языке. При систематической (тематической) группировке записи располагают в определенной логической последовательности в соответствии с принятой системой классификации. При хронологическом способе группировки записи располагают в хронологии выхода документов в свет.

ГОСТ Р 7.0.11-2011 не накладывает ограничений на форму цитирования источников. Ссылки на нумерованные списки можно приводить в квадратных скобках или в виде верхних индексов. В последнем случае номера источников в списке литературы также следует оформить в виде верхних индексов. Если список литературы не пронумерован, в тексте в квадратных скобках указывают фамилии авторов или название документа.

Приложения

Приложения оформляют в соответствии с ГОСТ Р 2.105, располагают в тексте диссертации или в ее конце и нумеруют сквозной нумерацией с остальной частью работы. Большие приложения выделяют в отдельный том и нумеруют отдельно.

Приложения должны быть перечислены в оглавлении в порядке ссылок на них в тексте диссертации. Отдельный том «Приложения» должен иметь титульный лист, аналогичный титульному листу основного тома диссертации с добавлением слова «Приложения», и самостоятельное оглавление. Наличие тома «Приложения» указывают в оглавлении первого тома диссертации.

Автореферат диссертации

Автореферат диссертации печатают типографским способом или на множительном аппарате в количестве, определяемом диссертационным советом. Автореферат включает в себя:

а) обложку автореферата;

на правах рукописи

Крылов Иван Андреевич

Векторные решения в задачах логистики

01.02.03 — перспективные технологии

Автореферат
диссертации на соискание ученой степени
доктора синергетических наук

Санкт-Петербург 1814

Императорская академия наук в Санкт-Петербурге

Консультант	**Jean de La Fontaine** membre de l'Académie française
Официальные оппоненты	**Лебедь Александр Иванович**, д. т. н., с. н. с, Национальный институт авиационных технологий, заведующий лабораторией.
	Рак Борис Семенович, д. б. н., профессор, ФГБНУ Всероссийский научно-исследовательский институт рыбного хозяйства и океанографии, главный научный сотрудник.
Ведущая организация	Институт экономики транспорта и транспортной политики национального исследовательского университета «Высшая школа экономики»

Защита состоится в ближайшем будущем на заседании диссертационного совета 24.1.080.04 при ИМЛИ РАН, расположенном по адресу: 121069, Москва, ул. Поварская, д. 25а

С диссертацией можно ознакомиться в библиотеке и на сайте https://ru.wikisource.org/wiki/Басни_(Крылов)

Автореферат разослан «__» _____ 2024 г.

Ученый секретарь диссертационного совета, к. фил. н. **Щука Анна Ивановна**

Рис. Г.4. Пример оформления автореферата диссертации

б) текст автореферата, включая:

1) общую характеристику работы,

2) основное содержание работы,

3) заключение;

в) список работ, опубликованных автором по теме диссертации.

На обложке автореферата приводят:

– статус документа — «на правах рукописи»;

– фамилию, имя и отчество диссертанта;

– название диссертации;

– шифр и наименование специальности по номенклатуре специальностей научных работников;

– искомую степень и отрасль науки;

– место и год написания автореферата диссертации.

На оборотной стороне обложки приводят следующие сведения:

– наименование организации, где выполнена диссертация;

– фамилию, имя, отчество, ученую степень, ученое звание научного руководителя (консультанта);

– фамилию, имя, отчество, ученую степень, ученое звание, место работы (организацию), должность официальных оппонентов;

– наименование ведущей организации;

– дату и время проведения защиты диссертации;

– шифр диссертационного совета;

– наименование и адрес организации, при которой создан совет;

– место ознакомления с диссертацией до защиты;

– дату рассылки автореферата диссертации;

– фамилию, имя, отчество ученого секретаря диссертационного совета.

Пример оформления обложки автореферата и ее оборотной стороны приведен на рис. Г.4.

Общая характеристика работы включает в себя следующие основные структурные элементы:

– актуальность темы исследования;

– степень ее разработанности;

– цели и задачи;

– научную новизну;

– теоретическую и практическую значимость работы;

– методологию и методы исследования;

– положения, выносимые на защиту;

– степень достоверности и апробацию результатов.

Основное содержание работы кратко раскрывает содержание глав диссертации. В заключении излагают итоги выполненного исследования, рекомендации и перспективы дальнейшей разработки темы. Библиографические записи оформляют в соответствии с требованиями ГОСТ Р 7.0.100.

Г.4. Верстка диссертации и автореферата

Требования к верстке диссертации заданы не столь жестко, как в случае отчета. Большую часть параметров верстки можно настроить по своему усмотрению. Для компиляции диссертации и автореферата удобно использовать общий файл, задать в нем класс документа, шрифты и общие настройки, а затем загрузить командами `\input` верстаемый текст с дополнительными настройками.

MainFile.tex

```
\documentclass[14pt]{extreport}
%\documentclass[11pt,twoside]{report}
\usepackage[russian]{babel}
\usepackage[T2A]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage{amssymb}
\usepackage{mathtools}
\allowdisplaybreaks
\usepackage{geometry}
\usepackage{indentfirst}
\usepackage{setspace}
\usepackage{caption}
\captionsetup{labelsep=endash}
\captionsetup[table]{singlelinecheck=off, justification=raggedright}
\captionsetup[figure]{justification=centering}
\usepackage[hidelinks]{hyperref}
\usepackage{array}
\usepackage{booktabs}
\usepackage{cite}
\usepackage[labeled,resetlabels]{multibib}
\newcites{A}{\mbox{Список работ,
                    опубликованных автором по теме диссертации}}
\bibliographystyle{ugost2008}
```

```
\bibliographystyleA{ugost2008}
\input{thesis}
%\input{abstract}
```

Диссертацию можно верстать через полтора интервала шрифтами Computer Modern в классе extreport шрифтами 14pt или в классе report шрифтами 12pt. Именно такой конфигурации соответствует приведенная преамбула, предназначенная для использования компилятора PDFLaTeX. Для верстки шрифтами Times New Roman, STIX Two Text или XITS (компилятор LaLaTeX) уберите загрузку пакетов inputenc и fontenc и скопируйте в преамбулу шрифтовые настройки отчета или те, что приведены на с. 305 или 310.

Рукопись диссертации имеет следующую структуру.

thesis.tex

```
\geometry{a4paper, left=25mm, right=10mm, top=20mm, bottom=20mm,
                headsep=5mm}
\onehalfspacing
\usepackage[imakeidx]{xindex}
\makeindex[options= -n -g -a -c abbreviations]
\usepackage{gost-thesis}
\begin{document}
\begin{center}
   ...   титульная страница   ...
\end{center}
\tableofcontents     %%% Оглавление
\chapter*{Введение}
   ...   текст введения   ...
\chapter{...}
   ...   основной текст отчета   ...   \index{...}   ...
\chapter*{Заключение}
   ...   текст заключения   ...
\chapter*{Список сокращений и условных обозначений}
%\input{\jobname.ind}
\bibliography{...}   %%% Список литературы
\bibliographyA{...}  %%% Список публикаций автора

\listoffgures        %%% Списки рисунков и таблиц
```

```
\appendix
\chapter{ПРИЛОЖЕНИЕ}
\section{...}
   ...   текст приложения   ...
\printindex
\end{document}
```

Пакет gost–thesis настраивает верстку разделов, оглавления, библиографии, списков и подписей рисунков и таблиц, а также окружения enumerate. Его загрузка имеет отличия при верстке диссертации и автореферата, поэтому она вынесена из общей преамбулы. Перечислим параметры загрузки, а затем обсудим их более подробно:

abstract — верстка автореферата;

NoClosingDot — печать номеров разделов без завершающей точки;

NoTitleBreak — печать наименований глав в строке с заголовками;

PristineLists — отмена настройки отбивок в списках;

plain — печать номеров страниц внизу по центру.

Текст диссертации делится на главы, разделы, подразделы, пункты и приложения. Главы и приложения формируют команды `\chapter{•}`. Они печатают заголовки в начале новой страницы, выводя перед ними наименование «Глава» или «Приложение» и номер. Предусмотрено три варианта печати:

Глава •
Заголовок из аргумента команды `\chapter`

Глава •. Заголовок из аргумента команды `\chapter`

Глава • Заголовок из аргумента команды `\chapter`

Первый используется по умолчанию. Параметр NoTitleBreak подавляет перенос заголовков на следующую строку и обеспечивает второй вариант, в котором номера глав завершаются точкой. Еще один параметр NoClosingDot убирает ее в заголовках глав (третий вариант) и всех вложенных разделов.

Команды `\chapter*{•}` настроены для оформления элементов диссертации, не являющихся разделами, таких как введение, заключение, библиография, словарь терминов, список сокращений и условных обозначений, списки рисунков и таблиц. Они печатают заголовки элементов и вносят их в оглавление.

Ссылки оформляет пакет cite, по умолчанию печатающий их в строке в квадратных скобках. Его настройки позволяют изменить формат вывода ссылок (см. разд. 8.4.1).

Оформление библиографии имеет следующие особенности. Нумерованный список литературы верстается в стиле ugost2008 компилятором bibtex. Номера источников в нем завершаются точками. Параметр NoClosingDot убирает точки и заключает номера в квадратные скобки. С помощью команд

`\BibLeft{`*левая скобка*`}` и `\BibRigh{`*правая скобка*`}`

можно заменить скобки любыми другими символами. Например, при настройке NoClosingDot установка `\def\BibLeft{}` и `\def\BibRigh{.}` уберет скобки и восстановит точки.

В диссертациях принято выделять работы автора в отдельный список. Это позволяет делать пакет multibib [102]. Он вводит команду

`\newcites{`*список суффиксов*`}{`*список заголовков*`}`,

которая для каждого *суффикса* • определяет команды `\bibliography`• и `\bibliographystyle`•, генерирующие дополнительные списки литературы.[8] В *списке заголовков* каждому суффиксу должен соответствовать заголовок библиографии, номера которой будут начинаться с данногом суффикса. Поясним сказанное примером. Команда `\newcites{A}{Публикации автора}` создаст команды `\bibliographyA` и `\bibliographystyleA`. В первой нужно указать стиль верстки библиографии, которую будет генерировать вторая команда. В начале списка литературы будет напечатан заголовок «Публикации автора», а источники получат номера A1, A2, A3 и т. д. Их будет печатать обычная команда `\cite`, в которой можно перечислять источники из любого списка. Например, так будет выглядеть ссылка, в которой цитируется три источника из библиографии, генерируемой командой `\bibliography`, и три из списка, создаваемого командой `\bibliographyA`: [1–3,A2–A4]. Порядок цитирования источников не важен, команда `\cite` отсортирует их общим списком. Настройки печати библиографии, заданные пакетом gost-thesis, применяются и к дополнительным спискам.

По умолчанию основной и дополнительные списки имеют сквозную нумерацию. Параметр resetlabels пакета multibib установит для дополнительных списков собственную нумерацию, а параметр labeled поставит перед их номерами суффиксы команды `\newcites`.

[8] Также создаются команды `\cite`•, `\nocite`•, которые вряд ли понадобятся для работы.

Для генерации дополнительных списков компилятор LaTeX сформирует файлы с именами •.aux (A.aux в нашем примере), где • — суффиксы команды \newcites . BibTeX не сможет обработать их автоматически, для каждого файла это придется делать вручную, как описано в в разд. А.2. BibTeX запишет сформированные списки в файлы •.bbl, и при последующих компиляциях LaTeX вставит их в документ.

Высота отбивки источников в списке литературы и записей в списках enumerate и itemize уменьшена вдвое по сравнению со стандартной. Это сделано с помощью длин \ItemSep и \TopSep, которые приравниваются к половине значений стандартных длин \itemsep и \topsep. Параметр PristineLists пакета gost-thesis отменяет эту настройку и восстанавливает стандартные отбивки. Изменение длин \ItemSep и \TopSep после загрузки пакета дает возможность настроить отбивки по своему усмотрению. Их величины удобно задавать в долях стандартных длин, например \ItemSep = 0.8\itemsep. Установка в нуль (\TopSep 0pt и т. п.), выполняемая параметром abstract, полностью убирает отбивки.

В соответствии с требованиями ГОСТ Р 7.0.11 пакет gost-thesis печатает номера страниц вверху, но в авторефератах, верстка которых регламентирована менее жестко, они обычно проставляются внизу. Это позволяет сделать параметр пакета plain.

abstract.tex

```
\geometry{a5paper, left=15mm, right=10mm, top=14mm, bottom=14mm,
                  headsep=4mm, footskip=8mm}
\singlespacing
\usepackage[abstract,plain]{gost-thesis}

\begin{document}
\begin{center}
   ...      Титульная страница      ...
\newpage
   ...      Оборотная страница      ...
   ...   Наименование организации   ...
\begin{tabular}{>\bfseries p{.35\linewidth} p{.6\linewidth}}
Научный руководитель  &    ...   \\[1ex]
Официальные оппоненты &    ...   \\[1ex]
           ...
\end{tabular}
\end{center}
   ...   Сведения о совете и защите   ...
```

```
\chapter*{Общая характеристика работы}
\textbf{Актуальность и степень разработанности темы исследования.}\par
            ...
\textbf{Цели и задачи иследования.}\par
            ...
\textbf{Теоретическая и практическая значимость работы.}\par
            ...
\textbf{Научная новизна.}\par
            ...
\textbf{Положения, выносимые на защиту.}
\begin{enumerate}
      ...
\end{enumerate}
\textbf{Степень достоверности и апробация результатов.}\par
            ...
\chapter*{Основное содержание работы}
   ...   Обзор текста диссертации   ...
\chapter*{Заключение}
   ...       Текст заключения       ...
\bibliography{...}   %%% Список литературы
\bibliographyA{...}  %%% Список публикаций автора
   ...   Выходные данные буклета автореферата  ...
\end{document}
```

Автореферат можно сверстать в классе report шрифтом 10pt или 11pt через один или полтора интервала. Для его верстки предназначен параметр abstract, устанавливающий формат страницы a5 с полями 15мм (левое) 10мм (правое), 14мм (верхнее и нижнее) и единичный межстрочный интервал, а также убирающий вертикальные отбивки элементов в списках.

Структура автореферата очень проста. Он не разбивается на разделы, а из специальных списков содержит лишь библиографию и список публикаций автора, поэтому параметр abstract адаптирует команды `\chapter*` и `\chapter*`, для оформления таких элементов автореферата, как «Общая характеристика работы», «Основное содержание работы», «Заключение», «Список цитируемых источников», «Список работ, опубликованных автором по теме диссертации» и т. п. Адаптированные команды печатают заголовки, вставляя над ними небольшую отбивку, не нумеруя их и не отправляя в оглавление.

gost-thesis.sty

```
\AtBeginDocument{
   \renewcommand\contentsname{Оглавление}
   \renewcommand\listfigurename{Список иллюстративного материала}
   \renewcommand\listtablename{Список таблиц}
   \renewcommand\bibname{Список литературы}
   \renewcommand\figurename{Рисунок}
   \renewcommand\tablename{Таблица}
   \renewcommand\chaptername{Глава}
   \thispagestyle{empty} }

\pagestyle{headings}
\def\@evenhead{\hfil\thepage\hfil}
\def\@oddhead{\hfil\thepage\hfil}

\counterwithin{figure}{chapter}
\counterwithin{table}{chapter}
\counterwithin{equation}{chapter}
\counterwithin{section}{chapter}
\setcounter{secnumdepth}{4}

\def\BibLeft{}
\def\BibRight{.}
\def\ClosingDot{.}
\let\TitleBreak\linebreak
\def\TuneLists{}
\def\@biblabel#1{\BibLeft#1\BibRight}

\IfPackageLoadedTF{multibib}
   {\expandafter\edef\csname @biblabel\@suffix \endcsname #1%
      {\noexpand\BibLeft\@suffix#1\noexpand\BibRight} }    {\relax}

\parindent 2.5em
\newlength\ParIndent \ParIndent \parindent
\newlength\ParSkip   \ParSkip   0.5\parskip
\newlength\ItemSep   \ItemSep   0.5\itemsep
\newlength\TopSep    \TopSep    0.5\topsep
\newlength\ParTopSep \ParTopSep 0.5\partopsep

\def\@seccntformat#1{\csname the#1\endcsname\ClosingDot\hspace{0.5em}}
\def\numberline#1{\hb@xt@\@tempdima{\ignorespaces#1\ClosingDot\hfil}}

\def\TitleFont{\normalfont \normalsize \bfseries \boldmath}
\def\ChapterFormat{\centering \TitleFont}

\def\ChapterTitle#1#2{
   \addcontentsline{toc}{chapter}
```

```
      {\protect\numberline{\chaptername\space\thechapter}
         \hphantom{\chaptername}#1}
   {\ChapterFormat\chaptername\space\thechapter\TitleBreak #2\par}
    \vskip\medskipamount \@afterheading }

\def\chapter{ \clearpage \secdef \@chapter \@element }

\def\@element#1{
   \phantomsection \addcontentsline{toc}{chapter}{#1}
   {\ChapterFormat #1 \par}
   \vskip\medskipamount \@afterheading }

\def\@chapter[#1]#2{
   \refstepcounter{chapter}
   \ChapterTitle{#1}{#2}
   \typeout{\chaptername\space\thechapter.}
   \addtocontents{lof}{\medskip}
   \addtocontents{lot}{\medskip} }

\def\section{\@startsection{section}{1}{0pt}
      {\bigskipamount}{\medskipamount}{\ChapterFormat}}

\def\subsection{\@startsection{subsection}{2}{0pt}
      {\bigskipamount}{\medskipamount}{\ChapterFormat}}

\def\subsubsection{\@startsection{subsubsection}{3}{0pt}
      {\medskipamount}{-0pt}{\indent\TitleFont}}

\def\l@chapter{\@dottedtocline{0}{0pt}{1.5em}}
\def\l@section{\@dottedtocline{1}{1.5em}{2em}}
\def\l@subsection{\@dottedtocline{2}{3.5em}{2.75em}}

\def\@makefntext#1{ \small
   \vspace{.5ex}\hspace{\ParIndent}
   \textsuperscript\@thefnmark #1}

\edef\appendix{ \unexpanded \expandafter {\appendix
  \def\thechapter{\Asbuk{chapter}}
  \def\chaptername{Приложение} } }

\edef\listoffigures{
   \unexpanded \expandafter {\listoffigures
      { \let\clearpage\relax \bigskip \listoftables} } }

\DeclareOption{NoClosingDot}{
   \def\ClosingDot{}
   \def\BibLeft{[}
   \def\BibRight{]} }

\DeclareOption{NoTitleBreak}{
    \def\TitleBreak{\ClosingDot\hspace{0.5em}} }
```

```
\DeclareOption{abstract}{
   \parindent 2.25em  \ParIndent \parindent
   \TopSep 0pt  \ItemSep 0pt  \parskip 0pt  \ParTopSep 0pt
   \def\chapter{ \secdef \@chapter \@element }
   \def\@element#1{\medskip {\ChapterFormat #1 \par}}
   \def\@chapter[#1]#2{\medskip {\ChapterFormat #2 \par}} }

\DeclareOption{plain}{
   \voffset-3mm
   \pagestyle{plain} }

\DeclareOption{PristineLists}{ \let\TuneLists\undefined }

\ProcessOptions\relax

\@ifundefined{TuneLists}\relax{
   \RequirePackage{calc}
   \def\SetLengths{\itemsep \ItemSep  \partopsep \ParTopSep
                   \parskip \ParSkip}
   \def\ReduceVspace{\vspace*{\TopSep-\topsep}}
   \let\SavedEnumerate\enumerate
   \renewenvironment{enumerate}{
      \begin{SavedEnumerate}
         \SetLengths \ReduceVspace }
      { \end{SavedEnumerate} }
   \let\SavedBibliography\thebibliography
   \renewenvironment{thebibliography}[1]{
   \begin{SavedBibliography}{#1}
       \SetLengths \ReduceVspace }
   { \end{SavedBibliography} }
   \let\SavedItemize\itemize
   \renewenvironment{itemize}{
      \begin{SavedItemize}
         \SetLengths \ReduceVspace }
      { \end{SavedItemize} } }
```

Бо̀льшая часть приведенного выше кода обсуждалась ранее для пакета gost–report.sty, поэтому рассмотрим лишь часть, содержащую новые команды и, прежде всего, те, что формируют инфраструктуру пакетов:

`\DeclareOption{`*параметр*`}{`*код*`}`,
`\DeclareOption*{`*код*`}`,
`\ExecuteOptions{`*список параметров*`}`,
`\ProcessOptions`, `\ProcessOptions*`.

Представленные выше команды определяют параметры загрузки пакетов. Первая создает *параметр*, при использовании которого будет выполняться *код*. Вторая задает *код*, исполняемый, когда пакет загружается с неизвестными параметрами. Код исполняют три последние команды. `\ExecuteOptions` делает это безотносительного того, использовались ли *параметры* при загрузке. Две другие команды получают список использованных параметров и обрабатывают лишь его, при этом `\ProcessOptions` исполняет код в порядке следования параметров в самом пакете, а `\ProcessOptions*` — в порядке их перечисления при загрузке.

Если для конструкций пакета требуются ресурсы других пакетов, их загружают командами

`\RequirePackage[`*параметры*`]{`*наименование пакета*`}[`*версия*`]` .

При загрузке можно указать список *параметров* и дату *версии* пакета в формате «год/месяц/день», например 2021/02/23. Пакет будет загружен, только если это не было сделано ранее. В нашем случае команда `\RequirePackage` загружает пакет calc, необходимый для вычисления длины пробела в команде `\ReduceVspace`.

Чтобы наладить взаимодействие с каким-либо пакетом, можно выяснить, загружен ли он. Для этого существуют команды

`\IfPackageLoadedTF{`*пакет*`}{`*код да*`}{`*код нет*`}` ,
`\IfPackageLoadedWithOptionsTF{`*пакет*`}{`*список параметров*`}{`*да*`}{`*нет*`}` .

Первая команда фиксирует только факт загрузки, а вторая проверяет, загружался ли пакет с определенным набором параметров. В зависимости от успешности проверки выполняется код предпоследнего или последнего аргумента. Пакет gost–report.sty использует команду `\IfPackageLoadedTF`, чтобы отследить загрузку пакета multibib и настроить печать номеров дополнительного списка литературы.

Перейдем к обсуждению ресурсов пакета gost–report.sty. Как уже говорилось, для оформления списков литературы он вводит команды `\BibLeft`, `\BibRight` и окружает ими номера источников в командах `\@biblabel` и `\@biblabel•`, где • — суффикс, определяемый пакетом multibib, находящийся в команде `\@suffix`. В начальных настройках команада `\BibLeft` пуста, а `\BibRight` содержит команду `\ClosingDot`, печатающую точку. Команда `\ClosingDot` добавлена также в команды `\@seccntformat` и `\numberline`, печатающие номера разделов в тексте и оглавлении. Параметр NoClosingDot делает команду `\ClosingDot` пустой, убирая тем самым точку из заголовков разделов, и помеща-

ет квадратные скобки в команды \BibLeft и \BibRight. В команду \ChapterTitle, формирующую заголовки глав, перед печатью номера командой \thechapter вставлена команда \chaptername, выводящая наименование «Глава» или «Приложение», а за номером следует команда \TitleBreak. В начальных настройках она является синонимом команды \linebreak, переносящей заголовки на следующую строку. Параметр NoTitleBreak помещает в нее команду \ClosingDot и небольшой пробел, разделяющий номера и заголовки, которые печатаются в одной строке.

Команды \pagestyle{headings}, \@evenhead и \@oddhead обеспечивают вывод номеров страниц вверху по центру в соотвествии с тебованиями ГОСТа. Параметр plain настраивает вывод номеров по центру снизу и освобождает под них место, сдвигая страницу на 3 мм вверх.

Для управления отбивками окружений thebibliography, enumerate и itemize вводится команда \TuneLists, а параметр PristineLists делает ее неопределенной. После обработки параметров команда \@ifundefined проверяет, существует ли команда \TuneLists и, если она определена, настраивает списки. Их отбивки корректируют команды \SetLengths и \ReduceVspace. Для этого каждое окружение сначала сохраняется в новое окружение, имя которого начинается словом «Saved...», а затем переопределяется с использованием этих команд. Первая присваивает длинам \itemsep, \parskip и \partopsep значения, уменьшающие отбивку элементов списков друг от друга и окружающего текста (см. рис. 8.5). Вторая компенсирует вертикальный пробел, автоматически вставляемый перед списками и предшествующим текстом. В начальных настройках длины \ParSkip, \ItemSep, \ParTopSep и \TopSep, корректирующие высоту отбивок, приравниваются половине своих стандартных значений. Параметр abstract, обращая их в нуль, убирает отбивки. Помимо установки длин данный параметр делает эквивалентными действия команд \chapter и \chapter*, настраивая их на печать заголовков полужирным шрифтом с небольшой верхней отбивкой.

Завершая обзор настроек пакета gost–thesis.sty, отметим, что они вносят минимальные коррективы в стандартные настройки LaTeX, все ресурсы которого могут быть задействованы в неизменном виде для верстки диссертации и автореферата.

Список литературы и интернет-источников

Данные об интернет-источниках, расположенных на официальном сайте TEX-ресурсов `www.ctan.org`, приведены по состоянию не позднее 01.03.2024. Даты обращения к ресурсам сайта размещения ГОСТов `https://protect.gost.ru` не позднее 01.02.2024. Источники, содержащие эти адреса, являются электронными ресурсами.

1. Кнут Дональд Е. Все про TEX. — Протвино : АО RDTEX, 1993. — С. 575. — ISBN: 5-900614-01-8.
2. Львовский С.М. Набор и верстка в пакете LATEX. — М. : МЦНМО, 2014. — С. 398. — ISBN: 978-5-4439-0239-5.
3. Котельников И., Чеботаев П. Настольная издательская система LATEX 2ε по-русски. — Новосибирск : Сибирский хронограф, 2004. — С. 496. — ISBN: 5–87550–195–2.
4. Гуссенс М., Миттельбах Ф., Самарин А. Путеводитель по пакету LATEX и его расширению LATEX 2ε. — М. : Мир, 1999. — С. 606. — ISBN: 5-03-003325-4.
5. Гуссенс М., Ратц С. Путеводитель по пакету LATEX и его web-приложениям. — М. : Мир, 2001. — С. 604. — ISBN: 5-03-003387-4.
6. Гуссенс М., Ратц С., Миттельбах Ф. Путеводитель по пакету LATEX и его графическим расширениям. — М. : Мир, 2002. — С. 621. — ISBN: 5-03-003388-2.
7. Грэтцер Г. Первые шаги в LATEX'е. — М. : Мир, 2000. — С. 172. — ISBN: 5-03-003366-1.
8. Роженко А. И. Искусство верстки в LATEX'е. — Новосибирск : ИВМиМГ СО РАН, 2005. — С. 398. — ISBN: 5-901548-25-6.
9. Балдин Е. М. Компьютерная типография LATEX. — СПб : БХВ-Петербург, 2008. — С. 304. — ISBN: 978-5-9775-0230-6.
10. Беляков С. Н., Палош В. Н., Садовский П. А. TEX для всех: Оформление учебных и научных работ в системе LATEX. — М. : Книжный дом «ЛИБРОКОМ», 2009. — С. 208. — ISBN: 978-5-397-00650-7.

11. American Mathematical Society, Downes Michael, updated by Beeton Barbara. — Standard document classes for LaTeX version 2e : 2024. — v1.4n. — www.ctan.org/pkg/classes.

12. Umeki Hideo. — The geometry package : 2020. — v5.9. — www.ctan.org/pkg/geometry.

13. REVTeX4.2 Author's Guide : 2019. — v4.2c. — www.ctan.org/pkg/revtex.

14. REVTeX4.2 Command and Options Summary : 2018. — v4.2c. — www.ctan.org/pkg/revtex.

15. elsarticle.cls – A better way to format your document : 2019. — v3.2. — www.ctan.org/pkg/elsarticle.

16. Mittelbach Frank. — The varioref package : 2020. — v1.6c. — www.ctan.org/pkg/varioref.

17. Carlisle David. — The indentfirst package : 2023. — v1.03. — www.ctan.org/pkg/indentfirst.

18. Kilfiger James, May Wolfgang. — The extsizes classes : 2007. — v1.4a. — www.ctan.org/pkg/extsizes.

19. Jeffrey Alan, Mittelbach Frank. — inputenc.sty : 2024. — v1.3d. — www.ctan.org/pkg/inputenc.

20. Schöpf Rainer, Raichle Bernd, Rowley Chris. — A new implementation of LaTeX's verbatim and verbatim* environments : 2001. — v2001/03/12. — www.ctan.org/pkg/verbatim.

21. Carlisle D. P. — Packages in the 'graphics' bundle : 2021. — v2020-03-05. — www.ctan.org/pkg/graphicx.

22. Fear Simon. — Publication quality tables in LaTeX : 2020. — v1.61803398. — www.ctan.org/pkg/booktabs.

23. van Oostrum Piet, Øystein Bache, Leichter Jerry. — The multirow, bigstrut and bigdelim packages : 2021. — v2.8. — www.ctan.org/pkg/multirow.

24. Jones David M. — The amsfonts package : 2013. — v3.01. — www.ctan.org/pkg/amsfonts.

25. American Mathematical Society, LaTeX3 project. — User's guide for the amsmath : 2020. — v2.1. — www.ctan.org/pkg/amsmath.

26. Mittelbach Frank, Schöpf Rainer, Downes Michael. — The amsbsy package : 1999. — v1.2d. — www.ctan.org/pkg/amsbsy.

27. Downes Michael. — The amsopn package : 2022. — v2.04. — www.ctan.org/pkg/amsopn.

28. Mittelbach Frank, Schöpf Rainer. — The amstext package : 2021. — v2.01. — www.ctan.org/pkg/amstext.

29. Carlisle David, Mittelbach Frank. — The bm package : 2019. — v1.2d. — www.ctan.org/pkg/bm.

30. Schmidt Walter. — The upgreek package for LaTeX 2ε : 2003. — v2.0. — www.ctan.org/pkg/upgreek.

31. Høgholm Morten, Madsen Lars, and the LaTeX3 project. — The mathtools package : 2021. — v1.27. — www.ctan.org/pkg/mathtools.

32. Carlisle David. — The delarray package : 2014. — v1.01. — www.ctan.org/pkg/delarray.

33. Publications Technical Group American Mathematical Society. — Using the amsthm package : 2017. — v2.20.3. — www.ctan.org/pkg/amsthm.

34. Mittelbach Frank. — An Extension of the LaTeX theorem environment : 2014. — v2.2c. — www.ctan.org/pkg/theorem.

35. Mittelbach Frank, Schöpf Rainer, Downes Michael. — The amscd package : 2017. — v2.1. — www.ctan.org/pkg/amscd.

36. Rose Kristoffer H. — XY-pic user's guide : 2013. — v3.8.9. — www.ctan.org/pkg/xypic.

37. Niederberger Clemens. — Chemformula : 2022. — v4.17. — www.ctan.org/pkg/chemformula.

38. Kuznetsov Alexey. — The chemformula-ru package : 2024. — v2024-03-01. — www.ctan.org/pkg/chemformula-ru.

39. Arseneau Donald. — The cite package: well formed numeric citations : 2015. — v2015/02/27. — www.ctan.org/pkg/cite.

40. Kotelnikov Igor, Sinev Leonid. — The GOST package : 2021. — v1.2l. — www.ctan.org/pkg/gost.

41. Kotelnikov Igor A. — The Russian Language in the babel system : 2021. — v1.3m. — www.ctan.org/pkg/babel-russian.

42. Sommerfeldt Axel. — Customizing captions of floating environments : 2023. — v2023-07-31. — www.ctan.org/pkg/caption.

43. ГОСТ Р 2.105-2019. Единая система конструкторской документации. Общие требования к текстовым документам. — М. : Стадартинформ, 2021. — https://protect.gost.ru/default.aspx/document1.aspx?control=31&baseC=6&page=0&month=9&year=-1&search=&id=237857.

44. ГОСТ 8.417-2002. Единицы величин. — М. : Стандартинформ, 2018. — https://protect.gost.ru/document.aspx?control=7&id=129858.

45. Schmidt Walter. — The icomma package for LaTeX 2ε : 2002. — v2.0. — www.ctan.org/pkg/icomma.

46. ГОСТ 7.32–2017. Отчето научно-исследовательской работе. Структура и правила оформления. — М. : Стадартинформ, 2018. — https://protect.gost.ru/document.aspx?control=7&id=218998.

47. ГОСТ Р 7.0.11-2011. Диссертация и автореферат диссертации. Структура и правила оформления. — М. : Стадартинформ, 2018. — https://protect.gost.ru/document.aspx?control=7&id=179727.

48. Braams Johannes, Carlisle David, Jeffrey Alan, Lamport Leslie, Mittelbach Frank, Rowley Chris, Schöpf Rainer. — The LaTeX 2_ε Sources : 2023. — 2023-06-01 Patch level 1. — www.ctan.org/pkg/source2e.

49. Mittelbach Frank. — LaTeX's hook management : 2024. — v1.1f. — https://tug.ctan.org/macros/latex/base/lthooks-code.pdf.

50. Carlisle David, Høgholm Morten. — The xspace package : 2014. — v1.13. — www.ctan.org/pkg/xspace.

51. Carlisle David. — The ifthen package : 2014. — v1.1c. — www.ctan.org/pkg/ifthen.

52. Thorup Kresten Krab, Jensen Frank, (and Chris Rowley). — The calc package. Infix notation arithmetic in LaTeX : 2020. — v4.3. — www.ctan.org/pkg/calc.

53. Arseneau Donald. — The ulem package: underlining for emphasis : 2019. — v v2019/11/18. — www.ctan.org/pkg/ulem.

54. Carlisle David. — The color package : 2022. — v1.3d. — www.ctan.org/pkg/color.

55. Kern Dr. Uwe. — Extending LaTeX's color facilities: the xcolor package : 2022. — v2.14. — www.ctan.org/pkg/xcolor.

56. Rahtz Sebastian, Oberdiek Heiko, Project the LaTeX3. — Hypertext marks in LaTeX: a manual for hyperref : 2023. — v7.01b. — www.ctan.org/pkg/hyperref.

57. Oberdiek Heiko. — The epstopdf package : 2020. — v2.11. — www.ctan.org/pkg/epstopdf-pkg.

58. Schlicht R. — The microtype package. Subliminal refinements towards typographical perfection : 2023. — v3.1a. — www.ctan.org/pkg/microtype.

59. Tobin Geoffrey, Fairbairns Robin. — The setspace package : 2022. — v2022/12/04. — www.ctan.org/pkg/setspace.

60. Bezos Javier. — The titlesec, titleps and titletoc Packages : 2023. — v2.16. — www.ctan.org/pkg/titlesec.

61. Arseneau Donald. — Chapterbib. multiple bibliographies in LaTeX : 2010. — v2010/09/05. — www.ctan.org/pkg/chapterbib.

62. Daly Patrick W. — Natural Sciences Citations and References : 2010. — v8.31b. — https://www.ctan.org/pkg/natbib.

63. Kime Philip, Wemheuer Moritz, Lehman Philipp. — The biblatex package. Programmable bibliographies and citations : 2022. — v3.18b. — www.ctan.org/pkg/biblatex.

64. Carlisle David. — The afterpage package : 2014. — v2014/10/28. — www.ctan.org/pkg/afterpage.

65. Oberdiek Heiko. — The pdflscape package : 2022. — v0.13. — www.ctan.org/pkg/pdflscape.

66. Mittelbach Frank. — An environment for multicolumn output : 2019. — v1.8y. — www.ctan.org/pkg/multicol.

67. Mittelbach Frank. — Footnotes in a multi-column layout : 2020. — v1.1f. — www.ctan.org/pkg/ftnright.

68. van Oostrum Piet. — The fancyhdr and extramarks packages : 2022. — v4.1. — www.ctan.org/pkg/fancyhdr.

69. Carlisle David. — The enumerate package : 2015. — v3.00. — www.ctan.org/pkg/enumerate.

70. American Mathematical Society, Downes Michael, updated by Beeton Barbara. — The amsart, amsproc, and amsbook document classes : 2020. — v2.20.6. — www.ctan.org/pkg/amsart.

71. American Mathematical Society, Beeton Barbara. — The amsmidx package : 2007. — v2.02. — www.ctan.org/pkg/amsmidx.

72. Mittelbach Frank, Carlisle David. — A new implementation of LaTeX's tabular and array environment : 2022. — v2.5g. — www.ctan.org/pkg/array.

73. Carlisle David. — The dcolumn package : 2014. — v1.06. — www.ctan.org/pkg/dcolumn.

74. Carlisle David. — The hhline package : 2020. — v2.04. — www.ctan.org/pkg/hhline.

75. Carlisle David. — The longtable package : 2021. — v4.17. — www.ctan.org/pkg/longtable.

76. Carlisle David. — The tabularx package : 2020. — v2.1.1c. — www.ctan.org/pkg/tabularx.

77. Carlisle David. — The colortbl package : 2018. — v1.0d. — www.ctan.org/pkg/colortbl.

78. Lyu Jianrui. — Tabularray. Typeset Tabulars and Arrays with LaTeX3 : 2024. — v2024A. — www.ctan.org/pkg/tabularray.

79. Pantigny F. — The package nicematrix : 2023. — v6.19a. — www.ctan.org/pkg/nicematrix.

80. Braams Johannes, Carlisle David, Jeffrey Alan, Lamport Leslie, Mittelbach Frank, Rowley Chris, Schöpf Rainer. — Standard LaTeXe packages makeidx and showidx : 2014. — v2014/09/29. — www.ctan.org/pkg/makeidx.

81. Gregorio Enrico. — The package imakeidx : 2016. — v1.3e. — www.ctan.org/pkg/imakeidx.

82. Voß Herbert. — Program and package xindex : 2023. — v0.55. — www.ctan.org/pkg/xindex.

83. Voß Herbert. — Program and package hvindex : 2017. — v0.04. — www.ctan.org/pkg/hvindex.

84. Talbot Nicola L. C. — User Manual for glossaries.sty v4.52 : 2022. — v4.52. — www.ctan.org/pkg/glossaries.

85. Talbot Nicola L. C. — The glossaries package v4.46: a guide for beginners : 2022. — v4.52. — www.ctan.org/pkg/glossaries.

86. Robertson Will, Hosny Khaled, Gesang Philipp, Wright Joseph. — The fontspec package. Font selection for XeLaTeX and LuaLaTeX : 2024. — v2.9a. — www.ctan.org/pkg/fontspec.

87. Robertson Will, Stephani Philipp, Wright Joseph, Hosny Khaled. — Experimental Unicode mathematical typesetting: The unicode-math package : 2023. — v0.8r. — www.ctan.org/pkg/unicode-math.

88. Robertson Will. — Symbols defined by unicode-math : 2023. — v0.8r. — www.ctan.org/pkg/unicode-math.

89. Tsolomitis Antonis. — The New Computer Modern FontFamily : 2024. — v5.2.0. — www.ctan.org/pkg/newcomputermodern.

90. Tsolomitis Antonis. — The fontsetup package : 2024. — v2.1.1. — www.ctan.org/pkg/fontsetup.

91. Zhao Yuansheng, Zeng Xiangdong. — Garamond-Math : 2022. — v2022-01-03. — www.ctan.org/pkg/garamond-math.

92. Morabity Mohamed El. — LaTeX support for Lato : 2019. — v3.3. — www.ctan.org/pkg/lato.

93. Voß Herbert. — OpenType math font Fira : 2023. — v2023/09/08. — www.ctan.org/pkg/firamath-otf.

94. American Mathematical Society. — Sample paper for the amsmath package. File name: testmath.tex : 1999. — v2.0. — www.ctan.org/pkg/libertinus-fonts.

95. Jones David M. — Unicode charts for STIX Two Math regular : 2021. — v2.12 b168. — www.ctan.org/pkg/stix2-otf.

96. Hoftich Michal. — The Lua-UCA library : 2021. — v0.1b. — www.ctan.org/pkg/lua-uca.

97. Kohm Markus. — Creating more than one index using splitidx and SplitIndex : 2016. — v1.2c. — www.ctan.org/pkg/splitindex.

98. Mittelbach Frank, Fairbairns Robin, Lemberg Werner and LaTeX project team. — LaTeX font encodings : 2016. — v2016/18/02. — www.ctan.org/pkg/fontenc.

99. ГОСТ Р 7.090-2016. Универсальная десятичная классификация. Структура, правила ведения и индексирования. — М. : Стандартинформ, 2020. — https://protect.gost.ru/document.aspx?control=7&id=205996.

100. ГОСТ Р 7.0.100-2018. Библиографическая запись. Библиографическое описание. Общие требования и правила составления. — М. : Стадартинформ, 2018. — https://protect.gost.ru/document.aspx?control=31&id=232175.

101. ГОСТ Р 7.0.80-2023. Библиографическая запись. Заголовок. Общие требования и правила составления. — М. : Российский институт стандартизации, 2023. — https://protect.gost.ru/document1.aspx?control=31&baseC=6&page=0&month=6&year=2023&search=&id=252942.

102. Hansen Thorsten. — The multibib package : 2008. — v1.4. — www.ctan.org/pkg/multibib.

Предметный указатель

Для удобства работы указатель разбит на несколько частей. Команды вывода многочисленных символов собраны в таблицах приложений:

В отдельные указатели сведены:

Символами {•} и [•] обозначены аргументы и параметры команд. Константы, счетчики и значения длин выделены рубленым шрифтом. Если стандартные классы устанавливают их однозначно, в скобках приведены их значения. Для динамических счетчиков указаны ассоциированные объекты.

Длины, средства оперирования шрифтами, а также ресурсы верстки библиографии, формул, таблиц и указателей помечены, соответственно, (д), (ш), (б), (ф), (т) и (у). Для ресурсов, определяемых пакетами, в скобках рубленым шрифтом приведены наименования пакетов. Такие же обозначения использованы и в указателях «Окружения» и «Пакеты».

Команды

A

D

E

N

O

S

T

Окружения

Символы {•} и [•] указывают на наличие аргументов и параметров, описание которых приведено в порядке их перечисления в окружениях.

Пакеты

Символы [•] обозначают наличие настроек, задаваемых при загрузке пакетов, список которых приведен после наименований пакетов. Настройки, описанные в книге, выделены рубленым курсивом. Перед номерами страниц в квадратных скобках приведены ссылки на документацию пакетов.

Список рисунков

Список таблиц

Оглавление

Алексей Владимирович Кузнецов

Основы LaTeX

Учебное пособие

Издание второе, исправленное и дополненное

Редактор М.В. Макарова
Оригинал-макет изготовлен А.В. Кузнецовым

Подписано в печать 02.08.2024. Формат 60×84 1/16.
Печ. л. 31.5. Уч.-изд. л. 31.5. Тираж 100 экз.
Изд. № 028-1.

Национальный исследовательский ядерный университет «МИФИ».
Типография НИЯУ МИФИ.
115409, Москва, Каширское ш., 31.