Не очень краткое введение в $\LaTeX\,2_\varepsilon$

Или LaTeX2e за 84 минуты

by Tobias Oetiker

Hubert Partl, Irene Hyna and Elisabeth Schlegl

Version 3.2, 21. September, 1998

Перевод: Б. Тоботрас, 7 октября 1998 г.

Благодарности

Многое из включенного в это введение материала исходит из австрийского введения в LaTeX 2.09, написанного на немецком:

Hubert Partl <partl@mail.boku.ac.at>
Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna <Irene.Hyna@bmwf.ac.at>
Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl <no email>
in Graz

Если Вы интересуетесь немецким вариантом документа, вы можете найти его версию, которую Jörg Knappen обновил для LaTeX 2_ε, по адресу `CTAN:info/lkurz`.

В процессе подготовки этого документа я просил о рецензиях в `comp.text.tex`. Я получил множество откликов. Читатели помогли улучшить эту книжку исправлениями, предложениями и материалами. Они старались помочь мне довести документ до его нынешнего состояния. Я хотел бы искренне их всех поблагодарить. Естественно, все ошибки, которые вы найдете в этой книжке, — мои. Случайно попавшее сюда правильно написанное слово наверняка обязано своим появлением репликой от одного из нижеперечисленных.

Rosemary Bailey, David Carlisle, Christopher Chin, Chris McCormack, Wim van Dam, David Dureisseix, Elliot, David Frey, Robin Fairbairns, Alexandre Guimond, Cyril Goutte, Greg Gamble, Neil Hammond, Rasmus Borup Hansen, Martien Hulsen, Werner Icking, Jakob, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Andrzej Kawalec, Alain Kessi, Christian Kern, Jörg Knappen, Kjetil Kjernsmo, Maik Lehradt, Martin Maechler, Claus Malten, Hubert Partl, John Refling, Mike Ressler, Brian Ripley, Young U. Ryu, Chris Rowley, Hanspeter Schmid, Craig Schlenter, Josef Tkadlec, Didier Verna, Fabian Wernli, Fritz Zaucker, Rick Zaccone, and Mikhail Zotov

Предисловие

LaTeX [1] — система верстки, ориентированная на производство научных математических документов высокого типографского качества. Система также вполне подходит для производства других видов документов, от простых писем до полностью сверстанных книг. LaTeX использует TeX[2] в качестве своего механизма верстки.

Это краткое введение описывает LaTeX 2_ε и должно быть достаточно для большинства применений LaTeX. Для полного описания системы LaTeX читайте [1, 3].

LaTeX работает на большинстве компьютеров, начиная с IBM PC или Mac, и кончая большими системами UNIX или VMS. В многих университетских сетях система уже установлена и готова к работе. Информация о том, как использовать локальную установку LaTeX, должна быть предоставлена в *Local Guide* [4]. Если у вас будут проблемы с началом работы, попросите о помощи того, кто предоставил вам эту книжку. Цель ее заключается *не* в том, чтобы обучить вас установке и настройке системы LaTeX, а в том, чтобы научить, как писать ваши документы так, чтобы они могли быть обработаны LaTeX.

Это Введение разбито на четыре главы:

Глава 1 рассказывает о базовой структуре документов LaTeX. Вы получите некоторое представление об истории LaTeX. После чтения этой главы вы должны получить грубое представление о LaTeX. Оно поможет вам объединить в единое целое сведения из остальных глав.

Глава 2 углубляется в детали верстки ваших документов. Она объясняет большинство важных команд и окружений LaTeX. После чтения этой главы вы сможет писать первые документы.

Глава 3 объясняет, как верстать формулы в LaTeX. Множество примеров поможет вам понять, как использовать эту, одну из сильных, сторон LaTeX. В конце этой главы вы найдете таблицы, перечисляющие большинство известных в LaTeX математических символов.

Глава 4 рассказывает про генерацию предметного указателя и библиографии, включение EPS графики, и про некоторые другие полезные расширения.

Глава 5 содержит потенциально опасные сведения о том, как менять стандартный макет документа. Она расскажет вам, как легко испортить красивый вывод LaTeX.

Важно читать главы последовательно. В конце концов, книжка не такая уж большая. По мере чтения вам встретится большое количество примеров. Читайте их внимательно.

Если вам понадобится любой, относящийся к LaTeX, материал, поищите его на одном из ftp архивов `CTAN`. Для США это — `ctan.tug.org`, для Германии — `ftp.dante.de`, для Великобритании — `ftp.tex.ac.uk`. Если вы не в одной из этих стран, выберите ближайший к вам архив.
Если вы хотите иметь LaTeX на вашем собственном компьютере, посмотрите на то, что что доступно по адресу `CTAN:systems`.

Если у вас есть мысли по поводу того, что сто́ит добавить, удалить или изменить в этом документе, пожалуйста, дайте мне знать. Я особенно заинтересован в откликах от новичков в LaTeX на тему того, какие части введения легко понимаемы, и что можно объяснить лучше.

Tobias Oetiker <oetiker@ee.ethz.ch>

Department of Electrical Engineering,
Swiss Federal Institute of Technology

Текущая версия этого документа доступна на
`CTAN:info/lshort`[1].

[1] Текущая версия перевода доступна на `http://xtalk.price.ru/tex`, а также на `CTAN`.

Оглавление

Список иллюстраций

Список таблиц

1 Это нужно знать

Первая часть этой главы содержит краткий обзор философии и истории LaTeX. Вторая часть главы фокусируется на основных структурах документов LaTeX. После чтения этой главы вы должны иметь общее представление о том, как работает LaTeX. В дальнейшем это поможет вам объединить всю новую информацию в единую картину.

1.1. Названия

1.1.1. TeX

TeX — это компьютерная программа, созданная Дональдом Кнутом (Donald E. Knuth) [2]. Она предназначена для верстки текста и математических формул. Кнут начал писать TeX в 1977 году из-за расстройства от того, чтó Американское Математическое Общество делало с его статьями в процессе их публикации. Где-то в 1974 году он даже прекратил посылать статьи: «просто мне было слишком больно смотреть на конечный результат». TeX, в том виде, в каком мы его сегодня используем, был выпущен в 1982 году и слегка улучшен с годами. Последние несколько лет TeX стал чрезвычайно стабилен. Кнут утверждает, что в нем практически нет ошибок. Номер версии TeX сходится к π и сейчас равен 3.14159.

TeX произносится как «тех». В среде `ASCII` TeX нужно писать как `TeX`.

1.1.2. LaTeX

LaTeX — макропакет, позволяющий авторам верстать и печатать их работы с высоким типографским качеством, при помощи заранее определенных, профессиональных макетов. LaTeX был написан Leslie Lamport [1]. В качестве механизма для верстки он использует TeX.

В 1994 году пакет LaTeX был обновлен командой LaTeX3 во главе с Frank Mittelbach. В нем были сделаны некоторые давно ожидавшиеся улучшения, и вновь объединены все варианты LaTeX, разошедшиеся с выпуска много лет назад версии LaTeX 2.09. Чтобы не путать эту новую версию со старой, она называется LaTeX 2_ε. Эта документация описывает именно LaTeX 2_ε.

LaTeX произносится как «лэйтех» или как «латех». Если вы ссылаетесь на LaTeX в `ASCII` окружении, пишите `LaTeX`. LaTeX 2_ε пишется как `LaTeX2e`.

Рисунок 1.1 на странице 2 показывает, как работают вместе TeX и LaTeX 2_ε. Он взят из `wots.tex` (автор Kees van der Laan).

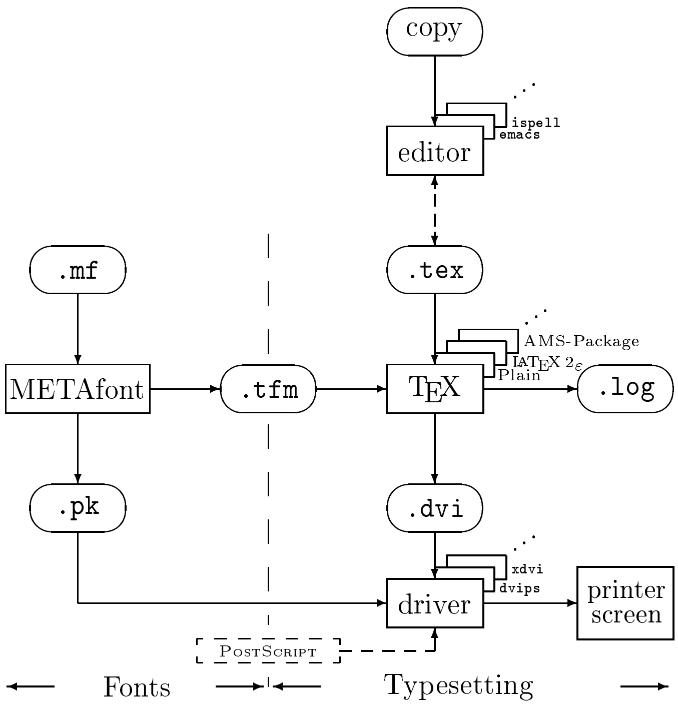


Рис. 1.1: Составляющее TEX системы

1.2. Основы

1.2.1. Автор, дизайнер и верстальщик

Для того, чтобы опубликоваться, авторы отдают свои рукописи в издательство. Затем один из дизайнеров издательства определяет макет документа (ширину столбцов, шрифты, интервалы выше и ниже заголовков и т.п.). Дизайнер записывает свои инструкции в рукописи и отдает ее верстальщику, который верстает книгу в соответствии с этими инструкциями.

Дизайнер–человек пытается понять, что автор имел в виду, когда писал свою рукопись. Он определяет заголовки глав, цитаты, примеры, формулы и прочее, исходя из своего профессионального опыта и из содержания рукописи.

В среде LaTeX, LaTeX берет на себя роль дизайнера книги, используя TeX в качестве верстальщика. Но LaTeX — это всего лишь программа, и, следовательно, нуждается в более четких инструкциях. Автор должен предоставить дополнительную информацию, описывающую логическую структуру своей работы. Эта информация записывается в текст в виде «команд LaTeX».

Это в корне отличается от WYSIWYG[1] подхода, принятого в большинстве современных текстовых процессоров, таких как *MS Word* или *Corel WordPerfect*. В этих приложениях авторы форматируют документ интерактивно в процессе набора текста на компьютере. В процессе работы они могут видеть на экране как будет выглядеть их работа, когда, в конце концов, она будет напечатана.

При использовании LaTeX обычно невозможно увидеть итоговую картину во время печатания текста. Ее, однако, можно посмотреть на экране после обработки файла LaTeX. Затем можно внести исправления перед собственно печатью.

1.2.2. Дизайн макета

Типографский дизайн — это профессия. Неопытные авторы часто допускают серьезные ошибки форматирования, предполагая, что дизайн книги — это большей частью вопрос эстетики: «если документ выглядит художественно, значит, он хорошо отдизайнен». Но, так как документ предназначен для чтения, а не для вывешивания в картинной галерее, удобство его чтения и понимания гораздо более важны, нежели красота. Например:

- Размер шрифта и нумерация заголовков должны выбираться с тем, чтобы сделать структуру глав и разделов ясной для читателя.

- Строка должна быть достаточно короткой, чтобы не напрягать глаза читателя, и достаточно длинной для красивого заполнения страницы.

С WYSIWYG системами авторы часто производят эстетически приятные документы со слабо выраженной или невыдержанной структурой. LaTeX предотвращает такие ошибки форматирования, заставляя автора объявлять *логическую* структуру

[1] What you see is what you get.

его документа. Затем уже LaTeX выбирает наиболее подходящий макет (раскладку) документа.

1.2.3. Преимущества и недостатки

Тема, часто обсуждаемая, когда люди из мира WYSIWYG встречаются с пользователями LaTeX, — «преимущества LaTeX перед нормальными текстовыми процессорами», или наоборот. Лучшее, что вы можете сделать, когда начинается такая дискуссия, — это пригнуться, так как она часто выходит из-под контроля. Однако, иногда вы не можете уклониться ...

Вот вам некоторое оружие. Основные преимущества LaTeX перед обычными текстовыми процессорами:

- Готовые профессионально выполненные макеты, делающие документы действительно выглядящими «как изданные».
- Удобно поддержана верстка математических формул.
- Пользователю нужно выучить лишь несколько понятных команд, задающих логическую структуру документа. Ему практически никогда не нужно возиться собственно с макетом документа.
- Легко изготавливаются даже сложные структуры, типа примечаний, оглавлений, библиографий и прочее.
- Для решения многих типографских задач, не поддерживаемых напрямую базовым LaTeX, есть свободно распространяемые дополнительные пакеты. Например, существуют пакеты для включения PostScript графики или для верстки библиографий в точном соответствии с конкретными стандартами. Многие из этих дополнительных компонент описаны в *The LaTeX Companion* [3].
- LaTeX поощряет авторов писать хорошо структурированные документы, так как именно так LaTeX и работает — путем спецификации структуры.
- TeX, форматирующее сердце LaTeX 2_ε, чрезвычайно мобилен и свободно доступен. Поэтому система работает практически на всех существующих платформах.

LaTeX имеет также и некоторые недостатки, но, кажется, мне трудно найти среди них заметные, хотя, я уверен, другие вам найдут их сотни `;-)`

- Хотя предопределенные макеты имеют множество настраиваемых параметров, создание полностью нового макета документа не очень просто и занимает много времени.[1]
- Очень сложно писать неструктурированные и неорганизованные документы.

[1] Говорят, что это — одна из основных целей будущей системы LaTeX3.

1.3. Исходные файлы LaTeX

Исходными данными для LaTeX являются обычный текстовый файл в `ASCII`. Его можно создать в любом текстовом редакторе. Он содержит текст документа вместе с командами, указывающими LaTeX, как верстать текст.

1.3.1. Пробелы

«Пустые» символы, такие, как пробел или табуляция, трактуются LaTeX одинаково, как «пробел». *Несколько последовательных* пустых символов трактуются как *один* «пробел». Пустые символы в начале строки обычно игнорируются, а единичный перевод строки воспринимается как «пробел».

Пустая строка между двух строк текста определяет конец абзаца. *Несколько* пустых строк трактуются так же, как *одна* пустая строка. Ниже приведен пример. Справа — текст из входного файла, слева — форматированный вывод.

Неважно, вставляете ли вы один или несколько пробелов между словами.
Пустая строчка начинает новый абзац.

```
Неважно, вставляете ли вы
один   или    несколько
пробелов между словами.

Пустая строчка начинает
новый абзац.
```

1.3.2. Спецсимволы

Следующие символы являются зарезервированными символами, которые либо имеют в LaTeX специальное значение, либо имеются не во всех шрифтах. Если вы введете их в текст напрямую, то они обычно не напечатаются, а заставят LaTeX сделать что–нибудь, вами вовсе не предусмотренное.

```
$ & % # _ { }  ~  ^  \
```

Как вы позже увидите, эти символы можно использовать в ваших документах, добавляя к ним префикс «\»:

$ & % # _ { }

```
\$ \& \% \# \_ \{ \}
```

Прочие символы, как и многие, многие другие, можно набрать специальными командами в математических формулах или как акценты. Знак «\» *нельзя* вводить, добавляя перед ним еще один, так как эта команда (\\) используется для разрыва строки[1].

[1]Вместо этого пользуйтесь командой `$\backslash$`. Она дает ‘\’.

1.3.3. Команды LaTeX

Команды LaTeX чувствительны к регистру и принимают одну из следующих двух форм:

- Они начинаются с символа backslash «\» и продолжаются именем, состоящим только из букв. Имена команд завершаются пробелом, цифрой или любой другой «не-буквой».
- Они состоят из «\» и ровно одного специального символа.

LaTeX игнорирует пробелы после команд. Если вы хотите получить пробел после команды, вы должны поместить или «{}» и пробел, или специальную команду пробела после имени команды. «{}» не дает LaTeX игнорировать все пробелы после имени команды.

Я слышал, что Кнут разделяет людей, работающих с TeX на TeXников и TeXпертов.
Сегодня — 7 октября 1998 г.

```
Я слышал, что Кнут разделяет
людей, работающих с \TeX{}
на \TeX{}ников и \TeX пертов.\\
Сегодня~--- \today
```

Некоторые команды нуждаются в параметре, который должен быть задан между фигурными скобками «{ }» после имени команды. Некоторые команды поддерживают необязательные параметры, которые добавляются после имени команды в квадратных скобках «[ ]». Следующий пример использует некоторые команды LaTeX. Не задумывайтесь над ними, они будут разъяснены позже.

Вы можете *положиться* на меня!

```
Вы можете \textsl{положиться}
на меня!
```

Пожалуйста, начните новую строчку прямо тут!
Спасибо!

```
Пожалуйста, начните новую
строчку прямо тут!\newline
Спасибо!
```

1.3.4. Комментарии

Когда в процессе обработки входного файла LaTeX встречает символ %, он игнорирует остаток текущей строки, возврат каретки и все пробелы в начале следующей строки. Это полезно для добавления в исходный файл замечаний, которые не будут выводиться на печать.

Это Spercalifragilisticexpialidocious

```
Это Spercal%
         ifragilist%
     icexpialidocious
```

Знаком % можно также пользоваться, чтобы разбить длинные строчки в тех местах, где не разрешаются пробелы или переводы строк.

1.4. Структура входного файла

Когда LaTeX 2ε обрабатывает входной файл, он ожидает от него следования определенной структуре. Так, каждый входной файл должен начитаться с команды

```
\documentclass{...}
```

Она указывает, документ какого типа вы собираетесь писать. После этого, вы можете включать команды, влияющие на стиль документа в целом, или загружать пакеты, добавляющие новые возможности в систему LaTeX. Для загрузки такого пакета используется команда

```
\usepackage{...}
```

Когда вся настройка закончена[1], вы начинаете тело текста командой

```
\begin{document}
```

Теперь вы вводите текст с командами LaTeX. В конце документа вы добавляете команду

```
\end{document}
```

Все, что следует после нее, LaTeX игнорирует.

Рис. 1.2 показывает содержимое минимального файла для LaTeX 2ε. Несколько более сложный входной файл дан на рис. 1.3[2].

1.5. Макеты документов

[1] Область между \documentclass и \begin{document} называется *преамбулой*.

[2] На самом деле, для того, чтобы документы могли содержать русские буквы, необходимо подключить пакет русификации LaTeX. Поскольку в настоящее время таких пакетов несколько, используемая вами русификация может отличаться от той, что использовалась при переводе. В этом случае вместо пакета babel вам, возможно, понадобится подключать что-то другое. Ожидается, что в будущем babel будет стандартным средством локализации TeX. Проконсультируйтесь с вашим *Local Guide* [4] или с администратором. —**прим. переводчика.**

```
\documentclass{article}
\usepackage[russian]{babel}
\begin{document}
Краткость~--- сестра таланта.
\end{document}
```

Рис. 1.2: Минимальный файл LaTeX

1.5.1. Классы документов

Первое, что LaTeX должен знать при обработке входного файла, это тип создаваемого автором документа. Он задается командой `\documentclass`.

`\documentclass[`*опции*`]{`*класс*`}`

Здесь *класс* определяет тип создаваемого документа. Таблица 1.1 перечисляет классы документов, рассматриваемые в этом введении. В состав LaTeX 2_ε входят дополнительные классы для других документов, включая письма и слайды. Параметр *опции* изменяет поведение класса документа. Опции должны разделяться запятыми. В таблице 1.2 перечислены самые употребительные опции стандартных классов документов.

Пример: Входной файл для документа LaTeX может начинаться строкой

```
\documentclass[11pt,twoside,a4paper]{article}
```

она заставляет LaTeX набирать документ как *статью*, с базовым размером шрифта *одиннадцать пунктов* и форматировать документ для *двусторонней* печати на бумаге *формата A4*.

1.5.2. Пакеты

В процессе написания вашего документа, вы, вероятно, обнаружите, что в некоторых областях базовый LaTeX не сможет решить ваши проблемы. Если вы захотите включить в документ графику, цветной текст или исходный код программы из внешнего

```
\documentclass[a4paper,11pt]{article}
\usepackage{latexsym}
\usepackage[russian]{babel}
\begin{document}
\author{Б.~Тоботрас}
\title{Минимализм}
\frenchspacing
\begin{document}
\maketitle
\tableofcontents
\section{\cyr{}Начало}
Вот тут и начинается моя замечательная статья.
\section{\cyr{}Конец}
\ldots{} а тут она кончается.
\end{document}
```

Рис. 1.3: Пример реалистичной журнальной статьи файла, вам нужно будет расширить возможности LaTeX. Такие расширения называются пакетами. Пакеты активизируются командой

`\usepackage[`*опции*`]{`*пакет*`}`

Где *пакет* — это имя пакета, а *опции* — список ключевых слов, включающих специальные свойства пакета. Некоторые пакеты включены в основную поставку LaTeX 2$_\varepsilon$ (см. таблицу 1.3). Другие предоставляются отдельно. Дополнительная информация об установленных у вас пакетах может содержаться в *Local Guide* [4]. Основной источник информации о пакетах LaTeX — это *The LaTeX Companion* [3]. Она содержит описания сотен пакетов вместе с информацией о том, как писать ваши собственные расширения для LaTeX 2$_\varepsilon$.

Таблица 1.1: Классы документов

`article` для статей в научных журналах, презентаций, коротких отчетов, программной документации, приглашений ...

`report` для более длинных отчетов, содержащих несколько глав, небольших книжек, диссертаций ...

`book` для настоящих книг

`slides` для слайдов. Использует большие буквы без засечек. Вместо этого можно использовать FoilTeX[a].

[a] `CTAN:macros/latex/packages/supported/foiltex`

Таблица 1.2: Опции классов документов

`10pt`, `11pt`, `12pt` Устанавливает размер основного шрифта документа. Если ни одна из этих опций не указана, подразумевается `10pt`.

`a4paper`, `letterpaper` ... Определяет размер листа. По умолчанию подразумевается `letterpaper`. Так же могут быть указаны `a5paper`, `b5paper`, `executivepaper` и `legalpaper`.

`fleqn` Выключные формулы будут выровнены влево, а не отцентрированы.

`leqno` Формулы нумеруются слева, а не справа.

`titlepage`, `notitlepage` Указывает, должна начинаться новая страница после заголовка документа или нет. По умолчанию класс `article` не начинает новую страницу, а `report` и `book` — начинают.

`twocolumn` Заставляет LaTeX набирать документ в два столбца.

`twoside, oneside` Выбирает одно- или двусторонний вывод. По умолчанию классы `article` и `report` используют односторонний вывод, класс `book` — двусторонний вывод. Заметьте, что опция `twoside` *не* заставляет ваш принтер на самом деле печатать с двух сторон.

`openright, openany` Делает главы начинающимися или только на правой странице, или на первой доступной. Это не работает с классом `article`, так как он ничего не знает о главах. Класс `report` по умолчанию начинает главы на следующей странице, а класс `book` — на правой.

Таблица 1.3: Некоторые из распространяемых с LaTeX пакетов

---

doc Позволяет документировать программы на LaTeX.
Описан в `doc.dtx` [a] и в *The LaTeX Companion* [3].

exscale Предоставляет масштабированные версии расширенных математических шрифтов.
Описан в `ltexscale.dtx`.

fontenc Указывает, какую кодировку шрифта должен использовать LaTeX.
Описан в `ltoutenc.dtx`.

ifthen Предоставляет команды вида 'если … , то выполнять … , иначе выполнять … '.
Описан в `ifthen.dtx` и в *The LaTeX Companion* [3].

latexsym Чтобы подключить шрифт специальных символов LaTeX, нужно использовать пакет `latexsym`. Описан в `latexsym.dtx` и в *The LaTeX Companion* [3].

makeidx Предоставляет команды для генерации указателей.
Описан в разделе 4.3 и в *The LaTeX Companion* [3].

syntonly Обрабатывает документ, не печатая его.
Описан в `syntonly.dtx` и в *The LaTeX Companion* [3]. Это удобно для быстрой проверки на ошибки.

inputenc Позволяет указать входную кодировку, такую как ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM code pages, Apple Macintosh, Next, ANSI-Windows или определяемую пользователем. Описан в `inputenc.dtx`.

---

[a] Этот файл должен быть установлен на вашей системе, и вы можете получить `dvi` файл, напечатав `latex doc.dtx` в любом каталоге, где вы имеете права на запись. То же самое относится ко всем прочим файлам, упомянутым в этой таблице.

1.5.3. Стили страницы

LaTeX поддерживает три предопределенных комбинации верхнего колонтитула и нижнего колонтитула — так называемые стили страницы. Параметр *стиль* команды

`\pagestyle{`*стиль*`}`

определяет, какой из них использовать. Предопределенные стили страницы перечислены в таблице 1.4.

Таблица 1.4: Предопределенные стили страницы LaTeX

`plain` печатает номера страниц внизу страницы в середине нижнего колонтитула. Этот стиль установлен по умолчанию.

`headings` печатает название текущей главы и номер страницы в верхнем колонтитуле каждой страницы, а нижний колонтитул остается пустым. (Этот стиль использован в данном документе.)

`empty` делает и верхние, и нижние колонтитулы пустыми.

Возможно сменить стиль текущей страницы командой

`\thispagestyle{`*стиль*`}`

Описание того, как создавать ваши собственные стили колонтитулов, смотрите в *The LaTeX Companion* [3], а также в разделе 4.4 на странице 57.

1.6. Большие проекты

При работе с большими документами вам может быть удобно разделить входной файл на несколько частей. LaTeX содержит две команды, которые помогают это делать.

`\include{`*файл*`}`

Эту команду можно использовать в теле документа, чтобы включить в него содержимое другого файла. Заметьте, что LaTeX начнет новую страницу, прежде, чем обрабатывать материал, содержащийся в *файле*.

Вторая команда может использоваться только в преамбуле. Она указывает LaTeX читать только некоторые из включенных (`\include`) файлов.

`\includeonly{`*filename*`,`*filename*`, … }`

После выполнения в преамбуле документа этой команды будут выполняться только те команды `\include`, чьи имена файлов перечислены в аргументе команды `\includeonly`.

Команда `\include` начинает верстку включаемого текста на новой странице. Это удобно при использовании `\includeonly`, потому что границы страниц не будут двигаться, даже когда некоторые включаемые файлы пропускаются. Иногда это неудобно, и в этом случае вы можете использовать команду

`\input{`*filename*`}`

Она просто включает содержимое указанного файла.

2 Верстка текста

После прочтения предыдущей главы вы должны представлять, из чего состоит документ LaTeX. Вторая глава заполняет эту структуру всем необходимым для производства реальных материалов.

2.1. Структура текста и языка

Основная цель написания текста — передать читателю идеи, информацию или знания. Читатель лучше поймет текст, если эти идеи структурированы, и намного лучше увидит и почувствует эту структуру, если типографская форма отражает логическую и смысловую структуру содержания.

LaTeX отличается от других систем верстки в том, что вам нужно лишь задавать ему логическую и смысловую структуру текста. Он затем выбирает типографскую форму текста в соответствии с «правилами», заданными в файле класса документа и в различных стилевых файлах.

Самый важный элемент текста в LaTeX (и в наборе вообще) — абзац. Он называется «элементом текста», потому что абзац является той типографской формой, в которую облекается одна связная мысль или идея. В следующих разделах вы узнаете, как вы можете вызвать перевод строки, например, при помощи \\ и начало абзаца, например, оставив пустую строку. Поэтому, если начинается новая мысль, должен начинаться новый абзац, а, если нет, — используйте переводы строки. Если вы сомневаетесь в вашем разбиении на абзацы, подумайте о вашем тексте как о носителе идей и мыслей. Если вы начали новый абзац, продолжая изложение той же мысли, — убирайте начало нового абзаца. Если на протяжении абзаца начинается совершенно новая мысль — разбивайте его на два.

Большинство людей совершенно не понимает важности разумного разбиения текста на абзацы. Многие не задумываются о значении разрыва абзацев или, особенно в LaTeX, начинают абзацы, не зная этого. Последнюю ошибку особенно легко сделать, когда в текст включаются уравнения. Посмотрите на следующие примеры и разберитесь, почему иногда в них используются пустые строчки (разрывы параграфов), а иногда — нет. (Если вы еще не понимаете всех команд достаточно, чтобы понять примеры, прочтите, пожалуйста, эту и следующую главы и затем вернитесь к примерам снова.)

… когда Эйнштейн ввел свою формулу

$$e = m \cdot c^2 \; , \tag{2.1}$$

которая является самой широко известной и самой мало понятной физической формулой.

```
\ldots когда Эйнштейн
ввел свою формулу
\begin{equation}
  e = m \cdot c^2 \; ,
\end{equation}
которая является самой
широко известной и самой
мало понятной физической
формулой.
```

… откуда следует закон тока Кирхгоффа:

$$\sum_{k=1}^{n} I_k = 0 \; . \tag{2.2}$$

Закон напряжения Кирхгоффа может быть выведен …

```
\ldots откуда следует
закон тока Кирхгоффа:
\begin{equation}
  \sum_{k=1}{n} I_k = 0 \; .
\end{equation}

Закон напряжения Кирхгоффа
может быть выведен \ldots
```

… который имеет несколько преимуществ.

$$I_D = I_F - I_R \tag{2.3}$$

является ядром совершенно другой модели транзистора. …

```
\ldots который имеет
несколько преимуществ.

\begin{equation}
  I_D = I_F - I_R
\end{equation}
является ядром совершенно
другой модели транзистора.
\ldots
```

Следующая, более маленькая, единица текста — предложение. В английском тексте после точки, завершающей предложение, ставится бо́льший пробел, чем после точки, стоящей после сокращения. LaTeX старается определить, какую из них вы имели в виду. Если он ошибается, вы должны ему подсказать. Это объяснено далее в этой главе.

Структура текста охватывает даже отдельные части предложений. Большинство языков использует очень сложные правила пунктуации, но во многих языках (включая немецкий и английский) вы расставите почти все запятые на места, просто помня, что они означают: короткую остановку в потоке изложения. Если вы не уверены в своих запятых, прочтите предложение вслух, останавливаясь на каждой запятой. Если где-то это прозвучит странно, уберите эту запятую, если вы почувствуете необходимость остановки в другом месте, вставьте запятую.[1]

Наконец, абзацы текста должны быть также логично структурированы на более

[1]Все это мало применимо для русского, но я решил не выбрасывать этих абзацев из перевода, вдруг кому-то поможет писать на английском, скажем. — **прим. переводчика**

высоком уровне, объединяясь в главы, разделы, и так далее. Однако, типографский эффект записи, например, `\section{The Структура текста и языка}` обычно достаточно очевиден, чтобы было сразу ясно, как использовать эти высокоуровневые структуры.

2.2. Разбиение на строки и страницы

2.2.1. Выровненные абзацы

Книги часто верстаются так, чтобы все строчки имели одинаковую длину. LaTeX вставляет необходимые разрывы строк и пробелы между словами, оптимизируя форматирование абзаца как целого. При необходимости он также переносит слова, которые не помещаются на строке. От класса документа зависит то, как верстаются абзацы. Обычно каждый абзац начинается с красной строки, а дополнительного интервала между ними нет. Подробности смотрите в разделе 5.3.2.

В некоторых случаях может быть необходимо указать LaTeX разорвать строку:

`\\` или `\newline`

начинают новую строку, не начиная нового абзаца.

`\\*`

запрещает, кроме того, разрыв страницы после вставленного разрыва строки.

`\newpage`

начинает новую страницу.

`\linebreak[`*n*`]`, `\nolinebreak[`*n*`]`, `\pagebreak[`*n*`]` и `\nopagebreak[`*n*`]`

разрывают строку, запрещают разрыв строки, разрывают страницу и запрещают разрыв страницы, соответственно. Необязательный аргумент n позволяет автору влиять на их действие. Он может быть равен числу от нуля до четырех. Устанавливая n в значение, меньшее 4, вы оставляете LaTeX возможность игнорировать вашу команду, если результат будет очень плох. Не путайте эти команды «разрыва (break)» с командами «начала (new)». Даже если вы зададите команду «разрыва», LaTeX попытается выровнять правую границу страницы и общую высоту страницы, как описано в следующей секции. Если вы действительно хотите начать «новую строчку», используйте соответствующую команду.

LaTeX всегда пытается производить наилучшее из возможных разбиений строк. Если он не может найти способ разбить строки в соответствии со своими стандартами, он позволяет одной строке выступать из абзаца вправо. LaTeX затем выводит диагностику («overfull hbox») во время обработки входного файла. Чаще всего это случается, когда LaTeX не может найти место для переноса слова.[1] Давая команду `\sloppy`, вы можете сказать, чтобы LaTeX несколько ослабил свои стандарты. Тогда он сможет предотвратить такие слишком длинные строки, увеличивая интервалы между словами — даже если конечный вывод будет не оптимален. В этом случае пользователь получит предупреждение («underfull hbox»). В большинстве случаев результат выглядит не очень хорошо. Команда `\fussy` действует в обратную сторону.

2.2.2. Переносы

LaTeX переносит слова, когда это необходимо. Если алгоритм переносов не находит правильных точек переноса, вы можете исправить положение, сказав TeX об исключении при помощи следующих команд.

Команда

`\hyphenation{`*список слов*`}`

вызывает перенос слов, перечисленных в ее аргументе, только в местах, отмеченных «`-`». Эта команда должна даваться в преамбуле входного файла, и должна содержать только слова, состоящие из обычных букв. Регистр букв игнорируется. Следующий пример разрешает переносы в слове «hyphenation», так же, как и в слове «Hyphenation», и запрещает переносы в словах «FORTRAN», «Fortran» и «fortran».

Пример:

```
\hyphenation{FORTRAN Hy-phen-a-tion}
```

Команда `\-` вставляет в слово выборочный перенос. Он также становится единственной разрешенной точкой переноса в этом слове. Эта команда в особенности полезна для слов, содержащих специальные символы (например, символы с акцентами), потому что LaTeX не переносит такие слова автоматически.

I think this is: supercalifragilisticexpiali-
docious

```
I think this is: su\-per\-cal\-%
i\-frag\-i\-lis\-tic\-ex\-pi\-%
al\-i\-do\-cious
```

Несколько слов можно удержать вместе на одной строке командой

`\mbox{`*текст*`}`

Она будет в любом случае сохранять свой аргумент вместе.

[1] Хотя LaTeX и дает вам предупреждение, такие строки не всегда легко найти. Если вы в команде `\documentclass` используете опцию `draft`, такие строки будут отмечены толстой черной линией на правых полях.

Номер моего телефона скоро сменится. Он будет 0116 291 2319.

```
Номер моего телефона скоро сменится.
Он будет \mbox{0116 291 2319}.
```

Параметр *имя файла* должен содержать имя файла.

```
Параметр
\mbox{\emph{имя файла}} должен
содержать имя файла.
```

2.3. Специальные буквы и символы

2.3.1. Знаки кавычек

Для набора кавычек вы *не должны* использовать знак `"`, как на пишущей машинке. При верстке существуют специальные знаки открывающей и закрывающей кавычек. В LaTeX используйте два знака `` ` `` в качестве открывающей кавычки и два знака `'` в качестве закрывающей.[1]

“Пожалуйста, нажмите клавишу ‘x’ ”

```
``Пожалуйста, нажмите
клавишу `x' ''
```

2.3.2. Тире и дефисы

LaTeX знает про четыре вида тире. Три из них вы можете получить различным числом последовательных знаков `-`. Четвертое на самом деле не тире вовсе, а математический знак минус:

мальчик-с-пальчик, X-лучи
страницы 13–67
да — или нет?
0, 1 и -1

```
мальчик-с-пальчик, X-лучи\\
страницы 13--67\\
да~--- или нет?\\
$0$, $1$ и $-1$
```

Эти тире называются так: `-` дефис, `--` короткое тире, `---` длинное тире и `$-$` знак минуса.

2.3.3. Многоточие (. . .)

На пишущей машинке точка или запятая занимают то же пространство, что и любая другая буква. Напечатанные в книге, эти символы занимают очень мало места и верстаются очень близко к предшествующей букве. Поэтому вы не можете ввести многоточие просто напечатав три точки, так как расстояние между ними будет

[1] При верстке обычно требуются две пары кавычек, на случай «„кавычек“ в кавычках». При верстке на русском языке обычно вместо “таких” и ‘таких’ кавычек используют «эти» и „эти“. Какими командами они задаются — зависит от используемой русификации, но обычно это `\flqq`, `\frqq` (вместо этой пары часто используют лигатуры `<<` и `>>`) и `\glqq`, `\grqq`. — **прим. переводчика**

неправильным. Кроме того, для многоточия есть специальная команда, называющаяся

`\ldots`

Не так ... а вот так: Нью-Йорк, Токио, Будапешт, ...	`Не так ... а вот так:\\` `Нью-Йорк, Токио, Будапешт, \ldots`

2.3.4. Лигатуры

Некоторые комбинации букв верстаются не просто набором разных букв друг за другом, а с использованием специальных символов[1].

ff fi fl ffi ... вместо ff fi fl ffi ...

Эти так называемые лигатуры могут быть запрещены вставкой `\mbox{}` между двумя соответствующими буквами. Это может быть необходимо для слов, образованных из двух частей.

Не “shelfful” а “shelfful”	``` Не ``shelfful''\\ ``` ``` а ``shelf\mbox{}ful'' ```

2.3.5. Акценты и специальные символы

LaTeX поддерживает использование акцентыакцентов и специальных символов из многих языков. Таблица 2.1 показывает всевозможные акценты в применении к букве о. Понятно, что ее месте могут быть и другие буквы.

Чтобы поместить знак акцента над буквами i или j, точки над ними должны быть удалены. Это достигается набором `\i` и `\j`.

Hôtel, naïve, élève, smørrebrød, !‘Señorita!, Schönbrunner Schloß Straße	``` H\^otel, na\"\i ve, \'el\`eve,\\ ``` ``` sm\o rrebr\o d, !`Se\~norita!,\\ ``` ``` Sch\"onbrunner Schlo\ss{} ``` ``` Stra\ss e ```

2.4. Поддержка иностранных языков

Если вам нужно писать документы на отличных от английского языках, то LaTeX должен быть соответствующим образом сконфигурирован в двух пунктах:

[1] Лигатуры широко используются при верстке текстов на английском языке. Насколько мне известно, при верстке на русском языке лигатур не используется. Однако, механизм лигатур TeX используется для генерации, например, кавычек и тире. — **прим. переводчика**.

1. Все генерируемые автоматически текстовые строки[1] должны быть переведены на другой язык. Для многих языков эти изменения достигаются использованием пакета babel (автор Johannes Braams).

2. LaTeX должен знать правила переноса для нового языка. Подключение правил переноса к LaTeX более сложно. Оно включает перестроение форматного файла с другими разрешенными образцами переноса. Ваш *Local Guide* [4] должен содержать больше информации об этом.

Если ваша система уже соответствующим образом сконфигурирована, вы можете активизировать пакет babel добавлением команды

```
\usepackage[язык]{babel}
```

после команды `\documentclass`. Какие *языки* поддерживает ваша система, также может быть указано в *Local Guide* [4]. `Babel` автоматически активизирует правильные правила переноса для выбираемых вами языков. Если форматный файл вашего LaTeX не поддерживает переносов для выбранного языка, `babel` будет работать, но запретит переносы, что негативно скажется на внешнем виде документа.

Для некоторых языков babel вводит новые команды, упрощающие ввод специальных символов. Например, немецкий язык содержит множество диакритических знаков (äöü). С использованием babel вы можете вводить ö, печатая `"o` вместо `\"o`.

Некоторые компьютерные системы позволяют вам вводить специальные символы прямо с клавиатуры. LaTeX может обрабатывать такие символы. Начиная с выпуска LaTeX 2_ε от декабря 1994г., поддержка нескольких входных кодировок является частью базовой поставки LaTeX 2_ε. Ознакомьтесь с пакетом inputenc. При использовании этого пакета вы должны понимать, что другие могут неправильно

[1] Содержание, Список иллюстраций, Библиография ...

Таблица 2.1: Акценты и специальные символы

ò	`` \`o ``	ó	`\'o`	ô	`\^o`	õ	`\~o`
ō	`\=o`	ȯ	`\.o`	ö	`\"o`	ç	`\c c`
ŏ	`\u o`	ǒ	`\v o`	ő	`\H o`	ǫ	`\c o`
ọ	`\d o`	o̲	`\b o`	o͡o	`\t oo`		
œ	`\oe`	Œ	`\OE`	æ	`\ae`	Æ	`\AE`
å	`\aa`	Å	`\AA`				
ø	`\o`	Ø	`\O`	ł	`\l`	Ł	`\L`
ı	`\i`	ȷ	`\j`	¡	`` !` ``	¿	`` ?` ``

видеть ваши входные файлы на своем компьютере из-за использования другой кодировки. Например, немецкий акцент ä на IBM PC кодируется как 132, а на некоторых Unix системах, использующих ISO-LATIN 1, — как 228. Поэтому используйте эту возможность с осторожностью.

Иной случай — с кодировкой шрифта. Она определяет, в какой позиции в пределах шрифта TEX находится каждая буква. Оригинальный TEX шрифт Computer Modern содержит только 128 символов из старого 7-битного набора символов ASCII. Когда требуются акцентированные символы, TEX создает их, комбинируя нормальный символ с акцентом. Несмотря на то, что результирующая верстка выглядит прекрасно, этот подход не дает автоматическим переносам работать внутри слов, использующих акцентированные символы.

К счастью, большинство современных поставок TEX содержат копию шрифтов EC. Эти шрифты выглядят, как шрифты Computer Modern, но содержат специальные символы для большинства акцентированных букв, используемых в европейских языках. Используя эти шрифты, вы можете улучшить переносы в не-английских документах. Шрифты EC активизируются включением в преамбуле вашего документа пакета fontenc.

```
\usepackage[T1]{fontenc}
```

2.5. Пробелы между словами

Для получения ровного правого края вывода LATEX вставляет различные интервалы между словами. В конце предложения он вставляет слегка больший интервал, делая текст более читабельным[1]. LATEX предполагает, что предложения заканчиваются точками, вопросительными или восклицательными знаками. Если точка следует за буквой в верхнем регистре, она не считается концом предложения, так как точки после букв верхнего регистра обычно используются для сокращений.

Любое исключение из этих предположений должно быть явно оговорено автором. Знак «\» перед пробелом дает в результате пробел, который не будет увеличен. Знак «~» дает пробел, который не может увеличиться и который, кроме того, запрещает разрыв строки. Команда \@ перед точкой указывает, что эта точка заканчивает предложение, несмотря на то, что стоит за буквой верхнего регистра.

Mr. Smith was happy to see her	`Mr.~Smith was happy to see her\\`
cf. Fig. 5	`cf.~Fig.~5\\`
I like BASIC. What about you?	`I like BASIC\@. What about you?`

[1] В соответствии с традициями верстки, принятыми в английском языке— **прим. переводчика**.

Дополнительный пробел после точек можно запретить командой

```
\frenchspacing
```

которая указывает LaTeX *не* вставлять пробела после точки более, чем после обычных символов. Это обычно для языков, отличных от английского, за исключением библиографий. Если вы используете `\frenchspacing`, команда `\@` не нужна.

2.6. Заголовки, главы и разделы

Чтобы помочь читателю ориентироваться в вашей работе, вы должны разделять ее на главы, разделы и подразделы. LaTeX поддерживает это специальными командами, принимающими в качестве аргумента заголовок раздела. Ваше дело — использовать их в надлежащем порядке.

Класс `article` включает следующие команды секционирования:

```
\section{...}           \paragraph{...}
\subsection{...}        \subparagraph{...}
\subsubsection{...}     \appendix
```

В классах `report` и `book` вы можете использовать две дополнительные команды:

```
\part{...}              \chapter{...}
```

Так как глав (chapters) в классе `article` нет, то статьи довольно легко добавлять в книгу в качестве глав. Интервалы между разделами, нумерация и размер шрифта заголовков устанавливаются LaTeX автоматически.

Две из команд секционирования — особенные:

- Команда `\part` не влияет на последовательность нумерования глав.
- Команда `\appendix` аргумента не имеет. Она просто начинает нумеровать главы буквами вместо цифр[1].

LaTeX создает оглавление, беря заголовки разделов и номера страниц из предыдущего прохода по документу. Команда

```
\tableofcontents
```

вставляет оглавление в то место, где она вызвана. Чтобы получить правильное оглавление, новый документ должен быть обработан LaTeX дважды. В особых случаях может быть необходим и третий проход. Когда это потребуется, LaTeX вас предупредит.

Все вышеперечисленные команды секционирования существуют также в вариантах со звездочкой. Такой вариант получается добавлением `*` к имени команды. Они

[1] В классе `article` меняется нумерация разделов.

генерируют заголовки разделов, которые не нумеруются и не включаются в оглавление. Например, команда `\section{Справка}` становится `\section*{Справка}`.

Обычно заголовки разделов появляются в оглавлении точно в том же виде, в каком они вводятся в тексте. Иногда это невозможно из-за того, что заголовок слишком длинен для оглавления. Элемент оглавления может в этом случае указываться необязательным аргументом перед собственно заголовком.

```
\chapter[Прочтите! Это замечательно]{Это~--- длинный,
                  длинный и очень нудный заголовок}
```

Титульный лист документа в целом генерируется при помощи команды

`\maketitle`

Его содержимое должно быть определено командами

`\title{...}`, `\author{...}` и `\date{...}`

до момента вызова `\maketitle`. Аргумент команды `\author` может содержать несколько имен, разделенных командами `\and`.

Пример некоторых из упомянутых команд может быть найден на иллюстрации 1.3 на странице 8.

Помимо описанных выше команд секционирования, LaTeX 2_ε вводит три дополнительных команды для использования с классом `book`.

`\frontmatter`, `\mainmatter` и `\backmatter`

Они полезны для деления вашей публикации. Команды изменяют заголовки глав и нумерацию страниц так, как это ожидается от книги.[1]

2.7. Перекрестные ссылки

В книгах, отчетах и статьях часто встречаются перекрестные ссылки на иллюстрации, таблицы и отдельные части текста. Для этого LaTeX предоставляет следующие команды:

`\label{`*метка*`}`, `\ref{`*метка*`}` и `\pageref{`*метка*`}`

где *метка* — выбранный пользователем идентификатор. LaTeX заменяет `\ref` номером раздела, подраздела, иллюстрации, таблицы или уравнения, где была использована соответствующая команда `\label`. `\pageref` печатает номер страницы,

[1] `\frontmatter` начинает вводную часть книги, `\mainmatter` — основную, `\backmatter` — заключающую. **— прим. переводчика**

соответствующей команде `\label`[1]. Так же, как и в случае с заголовками разделов, здесь также используются номера из предыдущего прохода.

Ссылка на этот раздел выглядит так: «см. раздел 17 на стр. 25.»

```
Ссылка на этот
раздел~\label{sec:this}
выглядит так: <<см.
раздел~\ref{sec:this} на
стр.~\pageref{sec:this}.>>
```

2.8. Сноски

Команда

`\footnote{`*текст сноски*`}`

печатает сноску внизу текущей страницы. Сноски всегда должны помещаться после слова, к которому они относятся.

Пользователи LaTeX часто употребляют сноски[a].

[a]Это — сноска.

```
Пользователи \LaTeX{}
часто употребляют
сноски\footnote{%
  Это~--- сноска.}.
```

2.9. Выделенные слова

В рукописи, напечатанной на машинке, `важные слова выделяются` `подчеркиванием`. В печатных изданиях эти слова выделяются *курсивом.* Команда для переключения на шрифт *выделения* называется

`\emph{`*текст*`}`

Ее аргументом является текст для выделения. Что на самом деле делает эта команда, зависит от контекста:

Если вы используете *выделение в уже выделенном тексте, то LaTeX использует* прямой *шрифт.*

```
Если вы используете
\emph{выделение в уже
выделенном тексте, то
\LaTeX{} использует
\emph{прямой} шрифт.}
```

Отметьте отличие между командами *выделения* и смены *шрифта*:

[1]Заметьте, что эти команды не знают, на что именно они ссылаются. `\label` просто сохраняет последний автоматически генерируемый номер.

Вы можете также выделить *текст, набрав его курсивом,* шрифтом без *засечек* или в стиле *пишущей машинки*.

```
\textit{Вы можете также
  \emph{выделить} текст,
  набрав его курсивом,}
\textsf{шрифтом без
  \emph{засечек}}
\texttt{или в стиле
  \emph{пишущей машинки}.}
```

2.10. Окружения

Для верстки специальных видов текста LaTeX определяет множество окружений для разных типов форматирования:

`\begin{`*название*`}` *текст* `\end{`*название*`}`

где *название* определяет окружение. Окружения можно вызывать внутри окружений, соблюдая порядок вызова и возврата:

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

В следующих разделах рассказывается обо всех важных окружениях.

2.10.1. Список, перечисление и описание

Окружение `itemize` подходит для простых списков, окружение `enumerate` — для нумерованных списков, а окружение `description` — для описаний.

1. Вы можете как угодно смешивать окружения списков:
   - Но это может смотреться глупо.
   - С минусом.
2. Поэтому помните:

   Глупые вещи не станут умнее от помещения в список.

   Умные вещи, однако, вполне можно представить списком.

```
\flushleft
\begin{enumerate}
\item Вы можете как угодно
смешивать окружения списков:
\begin{itemize}
\item Но это может смотреться
глупо.
\item[-] С минусом.
\end{itemize}
\item Поэтому помните:
\begin{description}
\item[Глупые] вещи не станут
умнее от помещения в список.
\item[Умные] вещи, однако,
вполне можно представить
списком.
\end{description}
\end{enumerate}
```

2.10.2. Выравнивание влево, вправо и по центру

Окружения `flushleft` и `flushright` форматируют абзацы, выровненные влево или вправо. Окружение `center` дает центрированный текст. Если вы не используете \\ для указания разрывов строк, LaTeX определит их автоматически.

Этот текст
выровнен влево. LaTeX не старается
сделать все строки одинаковой длины.

```
\begin{flushleft}
  Этот текст\\ выровнен влево.
  \LaTeX{} не старается сделать
  все строки одинаковой длины.
\end{flushleft}
```

Этот текст
выровнен вправо. LaTeX не старается
сделать все строки одинаковой длины.

```
\begin{flushright}
  Этот текст\\ выровнен вправо.
  \LaTeX{} не старается сделать
  все строки одинаковой длины.
\end{flushright}
```

В центре
Земли

```
\begin{center}
  В центре\\Земли
\end{center}
```

2.10.3. Цитаты и стихи

Окружение `quote` полезно для цитат, важных фраз и примеров.

Типографское правило для длины строки:

> Никакая строка не должна содержать больше 66 символов.
>
> Вот почему LaTeX делает такими широкими поля страниц.

Поэтому в газетах часто применяют набор в несколько колонок.

```
Типографское правило для
длины строки:
\begin{quote}
Никакая строка не должна
содержать больше 66~символов.

Вот почему \LaTeX{} делает
такими широкими поля страниц.
\end{quote}
Поэтому в газетах часто
применяют набор в
несколько колонок.
```

Существуют еще два похожих окружения: `quotation` и `verse`. Окружение `quotation` полезно для более длинных цитат, охватывающих несколько абзацев, потому что оно начинает абзацы с красной строки. Окружение `verse` используют для стихов, где важны разрывы строк. Строки разделяются при помощи \\ в конце строки и пустой строки после каждой строфы.

Я знаю только одно английское стихотворение наизусть: про Шалтая-Болтая:

> Humpty Dumpty sat on a wall:
> Humpty Dumpty had a great fall.
> All the King's horses and all the King's men
> Couldn't put Humpty together again.

```
Я знаю только одно английское
стихотворение наизусть: про
Шалтая-Болтая:
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\\
Humpty Dumpty had a great fall.\\
All the King's horses and all
the King's men\\
Couldn't put Humpty together
again.
\end{verse}
\end{flushleft}
```

2.10.4. Буквальное воспроизведение

Текст, заключенные между `\begin{verbatim}` и `\end{verbatim}` будет напрямую напечатан, как набранный на пишущей машинке, со всеми пробелами и возвратами каретки, без выполнения каких бы то ни было команд LaTeX.

Внутри абзаца аналогичную функцию выполняет команда

`\verb+`*текст*`+`

Здесь «`+`» — это только пример символа-ограничителя. Вы может использовать любой символ, кроме букв, «`*`» или пробела. Многие примеры на LaTeX в этом буклете набраны этой командой.

Команда `\ldots` …

```
10 PRINT "HELLO WORLD ";
20 GOTO 10
```

```
Команда \verb|\ldots| \ldots

\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

```
вариант␣␣␣окружения
verbatim␣␣␣␣␣␣␣␣␣со
звездочкой␣выделяет
пробелы␣␣␣в␣␣тексте
```

```
\begin{verbatim*}
вариант   окружения
verbatim         со
звездочкой выделяет
пробелы   в  тексте
\end{verbatim*}
```

Команду `\verb` тоже можно использовать аналогичным образом со звездочкой:

`вот␣␣␣так␣:-)␣`

```
\verb*|вот   так :-) |
```

Окружение `verbatim` и команду `\verb` нельзя использовать внутри параметров других команд.

2.10.5. Таблицы

Окружение `tabular` используют для верстки таблиц, возможно, с горизонтальными и вертикальными линиями. LaTeX автоматически определяет ширину столбцов.

Аргумент *спецификация* команды

`\begin{tabular}{`*спецификация*`}`

определяет формат таблицы. Используйте `l` для столбца текста, выровненного влево, `r` для текста, выровненного вправо и `c` для центрированного текста, `p{`*ширина*`}` для столбца, содержащего выровненный текст с переносом строк, и `|` для вертикальной линии.

Внутри окружения `tabular` знак «`&`» переходит к следующему столбцу, команда `\\` начинает новую строку, а `\hline` вставляет горизонтальную линию.

54	шестнадцатеричное
124	восьмеричное
1010100	двоичное
84	десятичное

```
\begin{tabular}{|r|l|}
\hline
54 & шестнадцатеричное \\
124 & восьмеричное \\
1010100 & двоичное \\
\hline \hline
84 & десятичное \\
\hline
\end{tabular}
```

Добро пожаловать в абзац в рамочке. Надеемся, вам всем тут понравится.

```
\begin{tabular}{|p{4.7cm}|}
\hline
Добро пожаловать в абзац в
рамочке. Надеемся, вам всем
тут понравится.\\
\hline
\end{tabular}
```

Разделитель столбцов можно задать конструкцией `@{...}`. Эта команда удаляет пробел между столбцами и заменяет его на то, что включено в фигурные скобки. Одно из частых использований этой команды показано ниже, при рассказе о проблеме выравнивания по десятичной точке. Другое возможно использование — для подавления ведущего пробела в таблице при помощи `@{}`:

нет ведущего пробела

```
\begin{tabular}{@{} l @{}}
\hline
нет ведущего пробела\\
\hline
\end{tabular}
```

ведущий пробел слева и справа

```
\begin{tabular}{l}
\hline
ведущий пробел слева и справа\\
\hline
\end{tabular}
```

Поскольку встроенный способ выровнять числовые столбцы по десятичной точке отсутствует[1], мы можем «обмануть» TEX и добиться этого при помощи двух столбцов: выровненной вправо целой части и выровненной влево дробной. Команда `@{.}` в строке `\begin{tabular}` заменяет нормальный пробел между столбцами просто на «.», давая эффект одного столбца, выровненного по десятичной точке. Не забудьте заменить в ваших числах точку на разделитель столбцов (`&`)! Метку столбца можно поместить над нашим числовым «столбцом» командой `\multicolumn`:

Выражение с π	Значение
π	3.1416
π^{π}	36.46
$(\pi^{\pi})^{\pi}$	80662.7

```
\begin{tabular}{c r @{.} l}
Выражение с $\pi$       &
\multicolumn{2}{c}{Значение} \\
\hline
$\pi$                 & 3&1416  \\
$\pi^{\pi}$           & 36&46   \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

2.11. Плавающие объекты

Большинство публикаций в наши дни содержат множество иллюстраций и таблиц. Эти элементы нуждаются в специальном обращении с ними, так как они не могут быть разбиты между страницами. Одним из выходов было бы начинать новую страницу каждый раз, когда встречается иллюстрация или таблица, слишком большая, чтобы поместиться на текущей странице. Этот подход привел бы к тому, что страницы оставались бы частично пустыми, что смотрится очень плохо.

Для решения этой проблемы любая иллюстрация или таблица, не умещающаяся на текущей странице, может 'плавать', перемещаясь на следующую страницу в процессе заполнения текстом текущей. LATEX предлагает для плавающих объектов два окружения, одно для таблиц и одно для иллюстраций. Чтобы полностью использовать их преимущества, важно примерно представлять, как LATEX обрабатывает плавающие объекты. Иначе они могут стать источником разочарования из-за того, что LATEX помещает их не туда, куда вы хотите.

Давайте вначале рассмотрим команды, предоставляемые LATEX для плавающих объектов.

Любой материал, включенный в окружения `figure` или `table`, трактуется как

[1]Если на вашей системе установлен комплект 'tools', обратите внимание на пакет dcolumn.

плавающий. Оба окружения имеют необязательный параметр

\begin{figure}[*спецификация размещения*] или \begin{table}[*спецификация размещения*]

называемый *спецификацией размещения*. Этот параметр используется для указания LaTeX, куда можно перемещать плавающий объект. *Спецификация размещения* конструируется путем собирания в строчку *ключей размещения плавающего объекта*. См. таблицу 2.2.

Например, таблицу можно начать следующей строкой:

```
\begin{table}[!hbp]
```

Спецификация размещения `[!hbp]` позволяет LaTeX разместить таблицу прямо по месту (`h`), или внизу той же страницы (`b`), или на выделенной странице (`p`), и все это — даже если это будет смотреться не так уж хорошо (`!`). Если никакой спецификации размещения не задано, стандартные классы предполагают `[tbp]`.

LaTeX размещает каждый встреченный плавающий объект в соответствии с заданной автором спецификацией. Если объект нельзя поместить на текущей странице, он откладывается, помещаясь в очередь иллюстраций или в очередь таблиц[1]. Когда начинается новая страница, LaTeX проверяет, можно ли заполнить специальную страницу плавающими объектами из очередей. Если нет, то первый объект из каждой очереди считается только что встретившимся в тексте: LaTeX снова пытается разместить их в соответствии с их спецификациями (за исключением ‘h’, что уже невозможно). Новые встреченные в тексте плавающие объекты помещаются в соответствующие очереди. LaTeX сохраняет порядок, в котором встретились плавающие объекты соответствующего типа. Поэтому иллюстрация, которую не удается разместить, отталкивает все дальнейшие иллюстрации к концу документа. Следовательно:

[1]Эти очереди подчиняются дисциплине *fifo*: ‘первым вошел — первым вышел’.

Таблица 2.2: Ключи размещения плавающего объекта

Ключ	Разрешает помещать объект ...
`h`	*здесь же*, в том самом месте текста, где он появился. Обычно используется для маленьких объектов.
`t`	*наверху* страницы
`b`	*внизу* страницы
`p`	на *специальной странице*, содержащей только плавающие объекты.
`!`	не рассматривать большинство внутренних параметров[a], которые могут предотвратить размещение этого объекта.

[a]Таких, как максимальное число плавающих объектов, разрешенных на одной странице

> Если LaTeX не размещает плавающие объекты, как вы этого ожидаете, то часто это только один объект устроил затор в одной из очередей.

После объяснения этих механизмов остается еще несколько замечания про окружения `table` и `figure`. Командой

`\caption{`*текст заголовка*`}`

вы можете задать заголовок для объекта. Увеличивающийся номер и строка «Рисунок» или «Таблица» добавляются LaTeX.

Две команды

`\listoffigures` и `\listoftables`

работают аналогично команде `\tableofcontents`, печатая список иллюстраций или таблиц, соответственно. В этих списках заголовки повторяются целиком. Если вы используете длинные заголовки, то вы должны предоставить их краткий вариант для включения в списки. Это делается помещением краткого варианта в квадратные скобки после команды `\caption`.

```
\caption[Короткий]{Дддддлллллллиииинннннныыыыыыыыыыыйййй}
```

При помощи `\label` и `\ref` можно делать ссылки из вашего текста на плавающий объект.

Следующий пример рисует квадрат и вставляет его в документ. Подобную технику можно использовать, чтобы оставить в документе место под изображения, которые вы вставите позже.

```
Рисунок~\ref{white} является примером Поп-Арта.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Пять на пять сантиметров} \label{white}
\end{figure}
```

В этом примере LaTeX будет *очень сильно* (`!`) стараться разместить иллюстрацию прямо *по месту* (`h`)[1]. Если это невозможно, он попытается разместить ее *внизу страницы* (`b`). Если ему не удастся поместить иллюстрацию на текущей странице, он выяснит, можно ли создать страницу плавающих объектов, содержащую эту иллюстрацию и, возможно, некоторые таблицы из очереди таблиц. Если для отдельной страницы материала еще не накопилось, LaTeX начинает новую страницу и снова рассматривает иллюстрацию, как если бы она только что появилась в тексте.

[1]Предполагая, что очередь иллюстраций пуста

В определенных случаях может быть необходимо использовать команду

`\clearpage` или даже `\cleardoublepage`

Она указывает LaTeX немедленно разместить все плавающие объекты, остававшиеся в очередях, и затем начать новую страницу. `\cleardoublepage`, помимо этого, начинает новую левостороннюю страницу.

Позже вы узнаете, как включать в ваши документы LaTeX рисунки в формате PostScript.

3 Набор математических формул

Вот теперь мы готовы! В этой главе мы встретимся с основной мощью TEX: математической версткой. Но имейте в виду, что эта глава дает только поверхностный обзор. Хотя для многих из вас изложенных здесь вещей будет достаточно, не отчаивайтесь, если вы не сможете найти решение, отвечающее нуждам верстки вашей математики. Весьма возможно, что ваша проблема решается в AMS-LaTeX[1] или в каком-нибудь ином пакете.

3.1. Общие сведения

LaTeX включает в себя специальный режим для верстки математики. Математический текст внутри абзаца вводится между `\(` и `\)`, между `$` и `$` или между `\begin{math}` или `\end{math}`.

Складывая a в квадрате с b в квадрате, получаем c в квадрате. Или излагая языком математики: $c^2 = a^2 + b^2$

```
Складывая $a$ в квадрате с
$b$ в квадрате, получаем
$c$ в квадрате. Или
излагая языком математики:
$c^{2}=a^{2}+b^{2}$
```

TeX произносится как $\tau\epsilon\chi$.

100 м^3 воды.

Это исходит от моего $\heartsuit$.

```
\TeX{} произносится как
$\tau\epsilon\chi$.\\[6pt]
100~м$^{3}$ воды.\\[6pt]
Это исходит от моего
$\heartsuit$.
```

Большие математические уравнения или формулы предпочтительнее «выключать», то есть верстать их на отдельных строчках. Для этого заключайте их между `\[` и `\]` или между `\begin{displaymath}` и `\end{displaymath}`. Это дает вам не нумерованные формулы. Если вы хотите, чтобы LaTeX их нумеровал, пользуйтесь окружением `equation`.

[1] CTAN:macros/latex/packages/amslatex

Складывая a в квадрате с b в квадрате, получаем c в квадрате. Или излагая языком математики:

$$c^{2}=a^{2}+b^{2}$$

И еще одна строчка

```
Складывая $a$ в квадрате с
$b$ в квадрате, получаем
$c$ в квадрате. Или
излагая языком математики:
\begin{displaymath}
  c^{2}=a^{2}+b^{2}
\end{displaymath}
И еще одна строчка
```

При помощи `\label` и `\ref` вы можете в тексте ссылаться на уравнение.

$$\epsilon > 0 \tag{3.1}$$

Из (3.1) выводим ...

```
\begin{equation}
  \label{eq:eps}
  \epsilon > 0
\end{equation}
Из (\ref{eq:eps})
выводим \ldots
```

Заметьте, что выражения в выключных формулах верстаются в другом стиле:

$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$

```
$\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}$
```

$$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

```
\begin{displaymath}
  \lim_{n \to \infty}
  \sum_{k=1}^n \frac{1}{k^2}
  = \frac{\pi^2}{6}
\end{displaymath}
```

Математический режим отличается от *текстового режима*. Например, в *математическом режиме*:

1. Большинство пробелов и возвратов каретки не принимаются во внимание, так как все пробелы либо выводятся из логики математических выражений, или должны в явном виде задаваться командами вроде `\,`, `\quad` или `\qquad`.

2. Пустые строчки недопустимы. Каждая формула занимает только один абзац.

3. Каждая буква считается именем переменной, и верстается в этом качестве. Если вы хотите в формулу ввести нормальный текст (нормальный прямой шрифт с нормальными пробелами), то вам нужно вводить его командами `\textrm{...}`.

$$\forall x \in \mathbf{R}: \qquad x^{2} \geq 0 \tag{3.2}$$

```
\begin{equation}
  \forall x \in \mathbf{R}:
  \qquad x^{2} \geq 0
\end{equation}
```

$$x^2 \geq 0 \qquad \text{для всех } x \in \mathbf{R} \tag{3.3}$$

```
\begin{equation}
  x^{2} \geq 0\qquad
  \textrm{для всех }x\in
  \mathbf{R}
\end{equation}
```

Математики бывают очень строги к используемым символам: здесь будет удобно использовать ‘ажурные полужирные символы’, которые получаются командой `\mathbb` из пакетов amsfonts или amssymb. Последний пример теперь выглядит так:

$$x^2 \geq 0 \qquad \text{для всех } x \in \mathbb{R}$$

```
\begin{displaymath}
  x^{2} \geq 0\qquad
  \textrm{для всех }x\in
  \mathbb{R}
\end{displaymath}
```

3.2. Группировка в математическом режиме

Большинство команд математического режима действует только на следующий символ. Так что, если вы хотите, чтобы команда влияла на несколько символов, вам нужно сгруппировать их вместе при помощи фигурных скобок: `{...}`.

$$a^x + y \neq a^{x+y} \tag{3.4}$$

```
\begin{equation}
  a^x+y \neq a^{x+y}
\end{equation}
```

3.3. Составляющие математической формулы

В этом разделе будут описаны наиболее важные команды, используемые в математической верстке. Детальный перечень команд для набора математических символов смотрите в разделе 3.9 на странице 46.

Строчные греческие буквы вводятся как `\alpha`, `\beta`, `\gamma`, ..., прописные буквы вводятся как `\Gamma`, `\Delta`, ... [1]

$\lambda, \xi, \pi, \mu, \Phi, \Omega$

```
$\lambda,\xi,\pi,\mu,%
\Phi,\Omega$
```

Верхние и нижние индексы вводятся при помощи символов «`^`» и «`_`».

$a_1 \qquad x^2 \qquad e^{-\alpha t} \qquad a^3_{ij}$
$e^{x^2} \neq {e^x}^2$

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

[1] В LaTeX 2_ε не определяется прописная «альфа», потому что она выглядит так же, как латинская «A». При новой кодировке математики это будет изменено.

Квадратный корень вводится как `\sqrt`, корень n-ной степени печатается при помощи `\sqrt[`n`]`. Размер знака корня выбирается LaTeX автоматически. Если нужен один только знак, используйте `\surd`.

$\sqrt{x} \qquad \sqrt{x^{2}+\sqrt{y}} \qquad \sqrt[3]{2}$
$\surd[x^2 + y^2]$

```
$\sqrt{x}$ \qquad
$\sqrt{ x^{2}+\sqrt{y} }$
\qquad $\sqrt[3]{2}$\\[3pt]
$\surd[x^2 + y^2]$
```

Команды `\overline` и `\underline` создают **горизонтальные линии** сразу над или под выражением.

$\overline{m+n}$

```
$\overline{m+n}$
```

Команды `\overbrace` и `\underbrace` создают длинные **горизонтальные фигурные скобки** сразу над или под выражением.

$\underbrace{a+b+\cdots+z}_{26}$

```
$\underbrace{ a+b+\cdots%
  +z }_{26}$
```

Для добавления к переменным знаков математических акцентов, таких, как маленькие стрелки или знака тильда, вы можете пользоваться командами, перечисленным в таблице 3.1. Широкие «шляпки» и тильды, охватывающие несколько символов, генерируются командами `\widetilde` и `\widehat`. Символ «`'`» дает знак производной..

$$y=x^{2}\qquad y'=2x \qquad y''=2$$

```
\begin{displaymath}
  y=x^{2}\qquad y'=2x
  \qquad y''=2
\end{displaymath}
```

Векторы часто указываются добавлением маленьких стрелок стрелки над переменной. Это делается командой `\vec`. Для обозначения вектора от A до B полезны две команды `\overrightarrow` и `\overleftarrow`.

$$\vec a\quad \overrightarrow{AB}$$

```
\begin{displaymath}
  \vec a\quad
  \overrightarrow{AB}
\end{displaymath}
```

Имена функций типа lg часто верстаются прямым шрифтом, а не курсивом, как переменные. Поэтому LaTeX содержит следующие команды для набора имен наиболее важных функций:

`\arccos`	`\cos`	`\csc`	`\exp`	`\ker`	`\limsup`	`\min`	`\sinh`
`\arcsin`	`\cosh`	`\deg`	`\gcd`	`\lg`	`\ln`	`\Pr`	`\sup`
`\arctan`	`\cot`	`\det`	`\hom`	`\lim`	`\log`	`\sec`	`\tan`
`\arg`	`\coth`	`\dim`	`\inf`	`\liminf`	`\max`	`\sin`	`\tanh`

$$\lim_{x \rightarrow 0} \frac{\sin x}{x}=1$$

```
\[\lim_{x \rightarrow 0}
\frac{\sin x}{x}=1\]
```

Для функции модуля функция модуля есть две команды: `\bmod` для бинарного оператора «$a \bmod b$» и `\pmod` для выражений вроде «$x \equiv a \pmod{b}$».

Двухъярусная **дробь** верстается командой `\frac{...}{...}`. Часто предпочтительнее ее форма с косой чертой 1/2, потому что она смотрится лучше при небольшом количестве 'дробного материала'.

$1\frac{1}{2}$ часа

$$\frac{x^{2}}{k+1}\qquad x^{\frac{2}{k+1}}\qquad x^{1/2}$$

```
$1\frac{1}{2}$~часа
\begin{displaymath}
  \frac{ x^{2} }{ k+1 }\qquad
  x^{ \frac{2}{k+1} }\qquad
  x^{ 1/2 }
\end{displaymath}
```

Для верстки биноминальных коэффициентов или аналогичных структур можно пользоваться или `{... \choose ...}`, или `{... \atop ...}`. Вторая команда дает тот же вывод, что и первая, но без скобок.

$$\binom{n}{k}\qquad {x \atop y+2}$$

```
\begin{displaymath}
{n \choose k}\qquad {x \atop y+2}
\end{displaymath}
```

Оператор интеграла печатает команда `\int`, **оператор суммы** — команда `\sum`. Верхние и нижние пределы указываются при помощи знаков «^» и «_», так же, как верхние и нижние индексы.

$$\sum_{i=1}^{n} \qquad \int_{0}^{\frac{\pi}{2}} \qquad$$

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\end{displaymath}
```

Для **скобок** и прочих ограничителей в TeX существует множество символов (скажем, $[\ \langle\ \|\ \ \updownarrow$). Круглые и квадратные скобки можно вводить соответствующими клавишами, фигурные скобки —`\{`, прочие ограничители — специальными командами (например, `\updownarrow`). Список доступных ограничителей смотрите в таблице 3.8 на странице 48.

$$a,b,c\neq\{a,b,c\}$$

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

Если вы поместите перед открывающим ограничителем команду `\left`, или перед закрывающим — `\right`, то TeX автоматически выберет правильный размер ограничителя. Заметьте, что вы должны каждый `\left` закрывать соответствующим `\right`, и что размер определяется корректно только если оба они набраны на одной строке. Если вы не хотите иметь правого ограничителя, используйте невидимый ограничитель ‘`\right.`’!

$$1 + \left(\frac{1}{1-x^{2}} \right)^3$$

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
    \right) ^3
\end{displaymath}
```

В некоторых случаях необходимо указать корректный размер математического ограничителя вручную, для этого есть команды `\big`, `\Big`, `\bigg` и `\Bigg`, служащие префиксами к большинству команд ограничителей[1].

$\Big((x+1) (x-1) \Big) ^{2}$

$\big(\Big(\bigg(\Bigg($ $\quad$ $\big\}\Big\}\bigg\}\Bigg\}$ $\quad$ $\big\|\Big\|\bigg\|\Bigg\|$

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

Чтобы ввести в формулу **три точки**, есть несколько команд. `\ldots` верстает точки на базовой линии, `\cdots` — центрированные. Кроме того, существуют команды `\vdots` для вертикальных и `\ddots` для диагональных точек. В разделе 3.5 вы найдете другой пример.

$$x_{1},\ldots,x_{n} \qquad x_{1}+\cdots+x_{n}$$

```
\begin{displaymath}
x_{1},\ldots,x_{n} \qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

3.4. Математические пробелы

Если выбранные TeX пробелы внутри формул неудовлетворительны, вы можете их подстраивать с использованием команд управления пробелами. Команды для маленьких пробелов: `\,` для $\frac{3}{18}$ quad (␣), `\:` для $\frac{4}{18}$ quad (␣) и `\;` для $\frac{5}{18}$ quad (␣). Экранированный символ пробела `\␣` дает средних размелов пробел, а `\quad` (␣) и `\qquad` (␣) дают большие пробелы. Размер `\quad` примерно соответствует ширине буквы ‘M’ в текущем шрифте. Команда `\!` производит отрицательный пробел размером $-\frac{3}{18}$ quad (␣).

[1] Эти команды не работают как ожидается, если используются команды смены размера шрифта, или если указана опция `11pt` или `12pt`. Для корректирования этого поведения пользуйтесь пакетами exscale или amsmath.

$$\int\!\!\!\int_{D} g(x,y)\, \mathrm{d}x\, \mathrm{d}y$$

вместо

$$\int\int_{D} g(x,y)\mathrm{d}x\mathrm{d}y$$

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int_{D} g(x,y)
  \, \ud x\, \ud y
\end{displaymath}
вместо
\begin{displaymath}
\int\int_{D} g(x,y)\ud x \ud y
\end{displaymath}
```

Заметьте, что ‘d’ в дифференциале обычно верстается прямым шрифтом.

$\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX включает другой способ тонкой подстройки пробелов между несколькими знаками интегралов: команды `\iint`, `\iiint`, `\iiiint` и `\idotsint`. С загруженным пакетом amsmath предыдущий пример можно набирать так:

$$\iint_{D} \, \mathrm{d}x \, \mathrm{d}y$$

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint_{D} \, \ud x \, \ud y
\end{displaymath}
```

Детали смотрите в электронном документе textmath.tex (распространяется с $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX) или в главе 8 “The LaTeX Companion”[1].

3.5. Вертикально расположенный материал

Для верстки **матриц** пользуйтесь окружением `array`. Его работа напоминает окружение `tabular`. Для разрыва строки используется команда `\\`.

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

Окружение `array` можно также использовать для верстки выражений, имеющих один большой ограничитель, подставляя «`.`» в качестве невидимого правого ограничителя:

[1] Можно получить на CTAN:info/ch8.*

$$y = \left\{ \begin{array}{ll} a & \textrm{если } d>c\\ b+x & \textrm{по утрам}\\ l & \textrm{остальное время дня} \end{array} \right.$$

```
\begin{displaymath}
y = \left\{ \begin{array}{ll}
 a & \textrm{если $d>c$}\\
 b+x & \textrm{по утрам}\\
 l & \textrm{остальное время дня}
  \end{array} \right.
\end{displaymath}
```

Для формул, занимающих несколько строк или для систем уравнений вместо `equation` пользуйтесь окружениями `eqnarray` и `eqnarray*`. В `eqnarray` каждая строка получает отдельный номер уравнения. В `eqnarray*` номера не ставятся.

Окружения `eqnarray` и `eqnarray*` работают наподобие таблицы из трех столбцов формата `{rcl}`, где средний столбец используется для знака равенства, или знака неравенства, или другого подходящего знака. Команда `\\` разбивает строки.

$$\begin{array}{rcl} f(x) & = & \cos x \qquad (3.5)\\ f'(x) & = & -\sin x \qquad (3.6)\\ \int_{0}^{x} f(y)dy & = & \sin x \qquad (3.7) \end{array}$$

```
\begin{eqnarray}
f(x) & = & \cos x      \\
f'(x) & = & -\sin x    \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

Заметьте, что по обеим сторонам средней колонки, знаков равенства, слишком много свободного места. Оно может быть уменьшено установкой `\setlength\arraycolsep{2pt}`, как в следующем примере.

Длинные уравнения не будут автоматически разбиваться на правильные части. Автор должен указать, где их разбивать и насколько выравнивать. Чаще всего для этого используют следующие методы:

$$\begin{array}{rcl} \sin x & = & x - \frac{x^{3}}{3!} + \frac{x^{5}}{5!} - \\ & & {}-\frac{x^{7}}{7!}+{}\cdots \qquad (3.8) \end{array}$$

```
{\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!}-{}
                    \nonumber\\
 & & {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}}
```

$$\begin{array}{l} \cos x = 1 - \frac{x^{2}}{2!} + \\ \qquad + \frac{x^{4}}{4!} - \frac{x^{6}}{6!} + \cdots \qquad (3.9) \end{array}$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
     -\frac{x^{2}}{2!} +{} }
                    \nonumber\\
 & & {}+\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!}+{}\cdots
\end{eqnarray}
```

Команда `\nonumber` заставляет LaTeX не генерировать номер для этого уравнения.

Такими методами может быть сложно получить правильно выглядящие вертикально выровненные уравнения; более мощную альтернативу предоставляет пакет amsmath.

3.6. Размер математического шрифта

В математическом режиме TEX выбирает размер шрифта в зависимости от контекста. Индексы, например, верстаются меньшим шрифтом. Если вы хотите добавить к уравнению обычный текст, не пользуйтесь командой `\textrm`, так как механизм переключения размера работать не будет, потому что `\textrm` временно выходит в текстовый режим. Чтобы оставить его работающим, используйте команду `\mathrm`[1]. Но имейте в виду, `\mathrm` будет хорошо работать только с короткими элементами. Пробелы по-прежнему не активны и акцентированные символы не работают[2].

$$2^{\mathrm{nd}} \quad 2^{\mathrm{nd}} \tag{3.10}$$

```
\begin{equation}
2^{\textrm{nd}} \quad
2^{\mathrm{nd}}
\end{equation}
```

Тем не менее, иногда вам может быть нужно указать LATEX точный размер шрифта. В математическом режиме размер устанавливается четырьмя командами:

`\displaystyle` (123), `\textstyle` (123), `\scriptstyle` ($\scriptstyle 123$) и `\scriptscriptstyle` ($\scriptscriptstyle 123$).

Смена стилей влияет также на способ изображения пределов.

$$\mathop{\mathrm{corr}}(X,Y)=\frac{\displaystyle\sum_{i=1}^n(x_i-\overline x)(y_i-\overline y)}{\displaystyle\biggl[\sum_{i=1}^n(x_i-\overline x)^2\sum_{i=1}^n(y_i-\overline y)^2\biggr]^{1/2}}$$

```
\begin{displaymath}
\mathop{\mathrm{corr}}(X,Y)=
 \frac{\displaystyle
   \sum_{i=1}^n(x_i-\overline x)
   (y_i-\overline y)}
  {\displaystyle\biggl[
 \sum_{i=1}^n(x_i-\overline x)^2
\sum_{i=1}^n(y_i-\overline y)^2
\biggr]^{1/2}}
\end{displaymath}
```

Это один из примеров, когда нам нужны ско́бки бо́льшие, чем предоставляемые стандартными `\left[ \right]`.

[1]В зависимости от используемой русификации, у вас могут не работать русские буквы внутри команды `\mathrm`, вместо которой может использоваться команда `\cyrmath` — **прим. переводчика**.

[2]При подключении пакета AMS-LATEX команда `\textrm` начинает работать с изменением размера.

3.7. Теоремы, законы, ...

При написании математических документов, вам, вероятно, нужен способ верстки «лемм», «определений», «аксиом» и аналогичных структур. LaTeX поддерживает это командами

```
\newtheorem{название}[счетчик]{текст}[раздел]
```

Аргумент *название* — это краткое ключевое слово, используемое для идентификации «теоремы». Аргументом *текст* вы определяете настоящее название «теоремы», под которым она будет печататься в документе.

Аргументы в квадратных скобках необязательны. Оба они используются для определения того, как нумеровать «теорему». Аргументом *счетчик* вы может указать *название* предварительно объявленной «теоремы». Новая «теорема» будет тогда нумероваться в той же последовательности. Аргумент *раздел* позволяет вам указать раздел, внутри которого вы хотите нумеровать вашу «теорему».

После использования в преамбуле документа команды `\newtheorem`, вы можете пользоваться следующими командами:

```
\begin{название}[текст]
Это интересная теорема.
\end{название}
```

На этом теории должно быть достаточно. Дальнейшие примеры должны развеять последнюю тень сомнений, и окончательно убедить вас, что окружение `\newtheorem` слишком сложно, чтобы его можно было понять:

Law 1 *Don't hide in the witness box*

Jury 2 (The Twelve) *It could be you! So beware and see law 1*

Law 3 *No, No, No*

```
% определения для
% преамбулы документа
\newtheorem{law}{Law}
\newtheorem{jury}[law]{Jury}
% в теле документа
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law \ref{law:box}\end{jury}
\begin{law}No, No, No\end{law}
```

Теорема «Jury» использует тот же счетчик, что и теорема «Law». Следовательно, она получит номер в последовательности с другими теоремами «Law». Аргумент в квадратных скобках указывает заголовок теоремы, или нечто аналогичное.

Murphy 3.7.1 *Если существует два или более способа сделать нечто, и один их этих способов может привести к катастрофе, то кто-то обязательно это сделает.*

```
\flushleft
\newtheorem{mur}{Murphy}[section]
\begin{mur}
Если существует два или
более способа сделать
нечто, и один их этих
способов может привести
к катастрофе, то кто-то
обязательно это сделает.
\end{mur}
```

Теорема «Murphy» получает номер, связанный с номером текущего раздела. Вы можете также использовать другую структурную единицу, например, главу или подраздел.

3.8. Полужирные символы

В LaTeX довольно непросто получить жирные символы; это, вероятно, сделано преднамеренно, потому что непрофессионалы слишком часто злоупотребляют ими. Команда смены шрифта `\mathbf` дает полужирные символы, но они обычные (прямые), тогда как математические символы обычно курсивные. Существует команда `\boldmath`, но *она может использоваться только вне математического режима.* То же относится и к символам.

$$\mu, M \qquad \mathbf{M} \qquad \boldsymbol{\mu}, \boldsymbol{M}$$

```
\begin{displaymath}
\mu, M \qquad \mathbf{M} \qquad
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

Заметьте, что запятая тоже полужирная, что может быть нежелательным.

Пакет amsbsy (включаемый пакетом amsmath) включает команду `\boldsymbol`.

$$\mu, M \qquad \boldsymbol{\mu}, \boldsymbol{M}$$

```
\begin{displaymath}
\mu, M \qquad
\boldsymbol{\mu}, \boldsymbol{M}
\end{displaymath}
```

3.9. Список математических символов

В следующих таблицах вы найдете все символы, известные обычно в *математическом режиме.*

Для доступа к символам, перечисленным в таблицах 3.12–3.16[1] в преамбуле документа должен быть загружен пакет amssymb, и в системе должны быть установлены математические шрифты AMS. Если пакеты и шрифты AMS в вашей системе не установлены, посмотрите на `CTAN:macros/latex/packages/amslatex`

Таблица 3.1: Акценты математического режима

$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\tilde{a}$	`\tilde{a}`	$\acute{a}$	`\acute{a}`
$\grave{a}$	`\grave{a}`	$\dot{a}$	`\dot{a}`	$\ddot{a}$	`\ddot{a}`	$\breve{a}$	`\breve{a}`
$\bar{a}$	`\bar{a}`	$\vec{a}$	`\vec{a}`	$\widehat{A}$	`\widehat{A}`	$\widetilde{A}$	`\widetilde{A}`

Таблица 3.2: Строчные греческие буквы

α	`\alpha`	θ	`\theta`	o	`o`	υ	`\upsilon`
β	`\beta`	ϑ	`\vartheta`	π	`\pi`	ϕ	`\phi`
γ	`\gamma`	ι	`\iota`	ϖ	`\varpi`	φ	`\varphi`
δ	`\delta`	κ	`\kappa`	ρ	`\rho`	χ	`\chi`
ϵ	`\epsilon`	λ	`\lambda`	ϱ	`\varrho`	ψ	`\psi`
ε	`\varepsilon`	μ	`\mu`	σ	`\sigma`	ω	`\omega`
ζ	`\zeta`	ν	`\nu`	ς	`\varsigma`		
η	`\eta`	ξ	`\xi`	τ	`\tau`		

Таблица 3.3: Прописные греческие буквы

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

[1]Эти таблицы были получены из `symbols.tex` (автор David Carlisle) и позже сильно изменены по совету Josef Tkadlec

Таблица 3.4: Бинарные отношения

Вы можете получить соответствующие отрицания добавлением перед следующими символами команды `\not`.

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` или `\le`	$\geq$	`\geq` или `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$	`\sqsubset` [1]	$\sqsupset$	`\sqsupset` [1]	$\Join$	`\Join` [1]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` или `\ne`

[1]Для доступа к этому символу пользуйтесь пакетом latexsym.

Таблица 3.5: Бинарные операторы

$+$	`+`	$-$	`-`		
$\pm$	`\pm`	$\mp$	`\mp`	$\triangleleft$	`\triangleleft`
$\cdot$	`\cdot`	$\div$	`\div`	$\triangleright$	`\triangleright`
$\times$	`\times`	$\setminus$	`\setminus`	$\star$	`\star`
$\cup$	`\cup`	$\cap$	`\cap`	$\ast$	`\ast`
$\sqcup$	`\sqcup`	$\sqcap$	`\sqcap`	$\circ$	`\circ`
$\vee$	`\vee` , `\lor`	$\wedge$	`\wedge` , `\land`	$\bullet$	`\bullet`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\diamond$	`\diamond`
$\odot$	`\odot`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\otimes$	`\otimes`	$\bigcirc$	`\bigcirc`	$\amalg$	`\amalg`
$\bigtriangleup$	`\bigtriangleup`	$\bigtriangledown$	`\bigtriangledown`	$\dagger$	`\dagger`
$\lhd$	`\lhd` [1]	$\rhd$	`\rhd` [1]	$\ddagger$	`\ddagger`
$\unlhd$	`\unlhd` [1]	$\unrhd$	`\unrhd` [1]	$\wr$	`\wr`

Таблица 3.6: Большие операторы

$\sum$	`\sum`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`	$\bigoplus$	`\bigoplus`
$\prod$	`\prod`	$\bigcap$	`\bigcap`	$\bigwedge$	`\bigwedge`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`			$\bigodot$	`\bigodot`
$\int$	`\int`	$\oint$	`\oint`			$\biguplus$	`\biguplus`

Таблица 3.7: Стрелки

$\leftarrow$	`\leftarrow` или `\gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\rightarrow$	`\rightarrow` или `\to`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\updownarrow$	`\updownarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\iff$	`\iff` (бóльший пробел)	$\leadsto$	`\leadsto` [1]

[1]Для доступа к этому символу пользуйтесь пакетом latexsym.

Таблица 3.8: Ограничители

$($	`(`	$)$	`)`	$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`
$[$	`[` или `\lbrack`	$]$	`]` или `\rbrack`	$\downarrow$	`\downarrow`	$\Downarrow$	`\Downarrow`
$\{$	`\{` или `\lbrace`	$\}$	`\}` или `\rbrace`	$\updownarrow$	`\updownarrow`	$\Updownarrow$	`\Updownarrow`
$\langle$	`\langle`	$\rangle$	`\rangle`	$\vert$		или \vert	$\Vert$	\\| или \Vert
$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`	$\lceil$	`\lceil`	$\rceil$	`\rceil`
$/$	`/`	$\backslash$	`\backslash`				

Таблица 3.9: Большие ограничители

$\lgroup$	`\lgroup`	$\rgroup$	`\rgroup`	$\lmoustache$	`\lmoustache`	$\rmoustache$	`\rmoustache`
$\arrowvert$	`\arrowvert`	$\Arrowvert$	`\Arrowvert`	$\bracevert$	`\bracevert`		

Таблица 3.10: Прочие символы

$\dots$	\dots	$\cdots$	\cdots	$\vdots$	\vdots	$\ddots$	\ddots
$\hbar$	\hbar	$\imath$	\imath	$\jmath$	\jmath	ℓ	\ell
$\Re$	\Re	$\Im$	\Im	$\aleph$	\aleph	$\wp$	\wp
$\forall$	\forall	$\exists$	\exists	$\mho$	\mho [1]	∂	\partial
$'$	'	$\prime$	\prime	$\emptyset$	\emptyset	∞	\infty
∇	\nabla	$\triangle$	\triangle	$\Box$	\Box [1]	$\Diamond$	\Diamond [1]
$\bot$	\bot	$\top$	\top	$\angle$	\angle	$\surd$	\surd
$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\clubsuit$	\clubsuit	$\spadesuit$	\spadesuit
$\neg$	\neg или \lnot	$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp

[1]Для доступа к этому символу пользуйтесь пакетом latexsym.

Таблица 3.11: Не-математические символы

Эти символы можно использовать и в текстовом режиме.

†	\dag	§	\S	©	\copyright
‡	\ddag	¶	\P	£	\pounds

Таблица 3.12: Ограничители AMS

$\ulcorner$	\ulcorner	$\urcorner$	\urcorner	$\llcorner$	\llcorner	$\lrcorner$	\lrcorner

Таблица 3.13: Буквы греческого и иврита AMS

$\digamma$	\digamma	$\varkappa$	\varkappa	$\beth$	\beth	$\daleth$	\daleth	$\gimel$	\gimel

Таблица 3.14: Бинарные отношения AMS

$\lessdot$	\lessdot	$\gtrdot$	\gtrdot	$\doteqdot$	\doteqdot или \Doteq
$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\risingdotseq$	\risingdotseq
$\eqslantless$	\eqslantless	$\eqslantgtr$	\eqslantgtr	$\fallingdotseq$	\fallingdotseq
$\leqq$	\leqq	$\geqq$	\geqq	$\eqcirc$	\eqcirc
$\lll$	\lll или \llless	$\ggg$	\ggg или \gggtr	$\circeq$	\circeq
$\lesssim$	\lesssim	$\gtrsim$	\gtrsim	$\triangleq$	\triangleq
$\lessapprox$	\lessapprox	$\gtrapprox$	\gtrapprox	$\bumpeq$	\bumpeq
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\Bumpeq$	\Bumpeq
$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless	$\thicksim$	\thicksim
$\lesseqqgtr$	\lesseqqgtr	$\gtreqqless$	\gtreqqless	$\thickapprox$	\thickapprox
$\preccurlyeq$	\preccurlyeq	$\succcurlyeq$	\succcurlyeq	$\approxeq$	\approxeq
$\curlyeqprec$	\curlyeqprec	$\curlyeqsucc$	\curlyeqsucc	$\backsim$	\backsim
$\precsim$	\precsim	$\succsim$	\succsim	$\backsimeq$	\backsimeq
$\precapprox$	\precapprox	$\succapprox$	\succapprox	$\vDash$	\vDash
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\Vdash$	\Vdash
$\Subset$	\Subset	$\Supset$	\Supset	$\Vvdash$	\Vvdash
$\sqsubset$	\sqsubset	$\sqsupset$	\sqsupset	$\backepsilon$	\backepsilon
$\therefore$	\therefore	$\because$	\because	$\varpropto$	\varpropto
$\shortmid$	\shortmid	$\shortparallel$	\shortparallel	$\between$	\between
$\smallsmile$	\smallsmile	$\smallfrown$	\smallfrown	$\pitchfork$	\pitchfork
$\vartriangleleft$	\vartriangleleft	$\vartriangleright$	\vartriangleright	$\blacktriangleleft$	\blacktriangleleft
$\trianglelefteq$	\trianglelefteq	$\trianglerighteq$	\trianglerighteq	$\blacktriangleright$	\blacktriangleright

Таблица 3.15: Стрелки AMS

$\dashleftarrow$	\dashleftarrow	$\dashrightarrow$	\dashrightarrow	$\multimap$	\multimap
$\leftleftarrows$	\leftleftarrows	$\rightrightarrows$	\rightrightarrows	$\upuparrows$	\upuparrows
$\leftrightarrows$	\leftrightarrows	$\rightleftarrows$	\rightleftarrows	$\downdownarrows$	\downdownarrows
$\Lleftarrow$	\Lleftarrow	$\Rrightarrow$	\Rrightarrow	$\upharpoonleft$	\upharpoonleft
$\twoheadleftarrow$	\twoheadleftarrow	$\twoheadrightarrow$	\twoheadrightarrow	$\upharpoonright$	\upharpoonright
$\leftarrowtail$	\leftarrowtail	$\rightarrowtail$	\rightarrowtail	$\downharpoonleft$	\downharpoonleft
$\leftrightharpoons$	\leftrightharpoons	$\rightleftharpoons$	\rightleftharpoons	$\downharpoonright$	\downharpoonright
$\Lsh$	\Lsh	$\Rsh$	\Rsh	$\rightsquigarrow$	\rightsquigarrow
$\looparrowleft$	\looparrowleft	$\looparrowright$	\looparrowright	$\leftrightsquigarrow$	\leftrightsquigarrow
$\curvearrowleft$	\curvearrowleft	$\curvearrowright$	\curvearrowright		
$\circlearrowleft$	\circlearrowleft	$\circlearrowright$	\circlearrowright		

Таблица 3.16: Отрицательные бинарные отношения и стрелки AMS

$\nless$	`\nless`	$\ngtr$	`\ngtr`	$\varsubsetneqq$	`\varsubsetneqq`
$\lneq$	`\lneq`	$\gneq$	`\gneq`	$\varsupsetneqq$	`\varsupsetneqq`
$\nleq$	`\nleq`	$\ngeq$	`\ngeq`	$\nsubseteqq$	`\nsubseteqq`
$\nleqslant$	`\nleqslant`	$\ngeqslant$	`\ngeqslant`	$\nsupseteqq$	`\nsupseteqq`
$\lneqq$	`\lneqq`	$\gneqq$	`\gneqq`	$\nmid$	`\nmid`
$\lvertneqq$	`\lvertneqq`	$\gvertneqq$	`\gvertneqq`	$\nparallel$	`\nparallel`
$\nleqq$	`\nleqq`	$\ngeqq$	`\ngeqq`	$\nshortmid$	`\nshortmid`
$\lnsim$	`\lnsim`	$\gnsim$	`\gnsim`	$\nshortparallel$	`\nshortparallel`
$\lnapprox$	`\lnapprox`	$\gnapprox$	`\gnapprox`	$\nsim$	`\nsim`
$\nprec$	`\nprec`	$\nsucc$	`\nsucc`	$\ncong$	`\ncong`
$\npreceq$	`\npreceq`	$\nsucceq$	`\nsucceq`	$\nvdash$	`\nvdash`
$\precneqq$	`\precneqq`	$\succneqq$	`\succneqq`	$\nvDash$	`\nvDash`
$\precnsim$	`\precnsim`	$\succnsim$	`\succnsim`	$\nVdash$	`\nVdash`
$\precnapprox$	`\precnapprox`	$\succnapprox$	`\succnapprox`	$\nVDash$	`\nVDash`
$\subsetneq$	`\subsetneq`	$\supsetneq$	`\supsetneq`	$\ntriangleleft$	`\ntriangleleft`
$\varsubsetneq$	`\varsubsetneq`	$\varsupsetneq$	`\varsupsetneq`	$\ntriangleright$	`\ntriangleright`
$\nsubseteq$	`\nsubseteq`	$\nsupseteq$	`\nsupseteq`	$\ntrianglelefteq$	`\ntrianglelefteq`
$\subsetneqq$	`\subsetneqq`	$\supsetneqq$	`\supsetneqq`	$\ntrianglerighteq$	`\ntrianglerighteq`
$\nleftarrow$	`\nleftarrow`	$\nrightarrow$	`\nrightarrow`	$\nleftrightarrow$	`\nleftrightarrow`
$\nLeftarrow$	`\nLeftarrow`	$\nRightarrow$	`\nRightarrow`	$\nLeftrightarrow$	`\nLeftrightarrow`

Таблица 3.17: Бинарные операторы AMS

$\dotplus$	`\dotplus`	$\centerdot$	`\centerdot`	$\intercal$	`\intercal`
$\ltimes$	`\ltimes`	$\rtimes$	`\rtimes`	$\divideontimes$	`\divideontimes`
$\Cup$	`\Cup` или `\doublecup`	$\Cap$	`\Cap` или `\doublecap`	$\smallsetminus$	`\smallsetminus`
$\veebar$	`\veebar`	$\barwedge$	`\barwedge`	$\doublebarwedge$	`\doublebarwedge`
$\boxplus$	`\boxplus`	$\boxminus$	`\boxminus`	$\circleddash$	`\circleddash`
$\boxtimes$	`\boxtimes`	$\boxdot$	`\boxdot`	$\circledcirc$	`\circledcirc`
$\leftthreetimes$	`\leftthreetimes`	$\rightthreetimes$	`\rightthreetimes`	$\circledast$	`\circledast`
$\curlyvee$	`\curlyvee`	$\curlywedge$	`\curlywedge`		

Таблица 3.18: Прочие символы AMS

$\hbar$	`\hbar`	$\hslash$	`\hslash`	$\Bbbk$	`\Bbbk`
$\square$	`\square`	$\blacksquare$	`\blacksquare`	$\circledS$	`\circledS`
$\vartriangle$	`\vartriangle`	$\blacktriangle$	`\blacktriangle`	$\complement$	`\complement`
$\triangledown$	`\triangledown`	$\blacktriangledown$	`\blacktriangledown`	$\Game$	`\Game`
$\lozenge$	`\lozenge`	$\blacklozenge$	`\blacklozenge`	$\bigstar$	`\bigstar`
$\angle$	`\angle`	$\measuredangle$	`\measuredangle`	$\sphericalangle$	`\sphericalangle`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`	$\backprime$	`\backprime`
$\nexists$	`\nexists`	$\Finv$	`\Finv`	$\varnothing$	`\varnothing`
$\eth$	`\eth`	$\mho$	`\mho`		

Таблица 3.19: Математические алфавиты

Пример	Команда	Требуемый пакет
ABCdef	`\mathrm{ABCdef}`	
ABCdef	`\mathit{ABCdef}`	
$\mathnormal{ABCdef}$	`\mathnormal{ABCdef}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	
$\mathscr{ABC}$	`\mathcal{ABC}`	mathrsfs
$\mathcal{ABC}$	`\mathcal{ABC}`	eucal с опцией: mathcal или
	`\mathscr{ABC}`	eucal с опцией: mathscr
$\mathfrak{ABCdef}$	`\mathfrak{ABCdef}`	eufrak
$\mathbb{ABC}$	`\mathbb{ABC}`	amsfonts или amssymb

4 Специальные возможности

При сборке большого документа LaTeX поможет вам некоторыми специальными функциями, например, генерацией предметного указателя, управлением библиографией и тому подобными. Более подробное описание специальных возможностей и расширений LaTeX находится в *LaTeX Manual* [1] и в *The LaTeX Companion* [3].

4.1. Включение EPS графики

LaTeX имеет базовые средства для работы с плавающими объектами, такими, как иллюстрациями и таблицами, при помощи окружений `figure` и `table`.

Существует также несколько способов создавать собственно графику средствами базового LaTeX или его расширений. К сожалению, большинство пользователей находит их трудными для понимания. Дополнительная информация приведена в *The LaTeX Companion* [3] и в *LaTeX Manual* [1].

Более простой метод получения графики в документе — это создавать ее специализированными программными пакетами[1] и включать в документ готовую графику. Пакеты LaTeX предлагают множество способов это делать. В этом введении обсуждается только использование графики в формате Encapsulated PostScript (EPS), поскольку это довольно просто делается и широко распространено. Чтобы использовать картинки в формате EPS, вам нужно использовать для вывода PostScript принтер[2].

Хороший набор команд включения графики входит в пакет graphicx (автор D. P. Carlisle). Он является частью целого семейства пакетов, называющегося комплектом “graphics”[3].

В предположении, что вы работаете в системе с доступным для вывода PostScript принтером, и с установленным пакетом graphicx, для включения картинки в ваш документ можете использовать следующую пошаговую инструкцию:

1. Экспортируйте картинку из вашей графической программы в формате EPS.[4]

[1] Такими, как XFig, CorelDraw!, Freehand, Gnuplot, ...

[2] Другая возможность вывода PostScript заключается в использовании программы GhostScript, доступной с `CTAN:support/ghostscript`.

[3] `CTAN:macros/latex/packages/graphics`.

[4] Если ваша программа не имеет экспорта в EPS, можно попробовать установить драйвер PostScript принтера (скажем, какого-нибудь Apple LaserWriter) и печатать в файл через этот драйвер. Если вам повезет, в этом файле будет EPS. Заметьте, что EPS обязан содержать не

2. В преамбуле документа загрузите пакет graphicx при помощи

   `\usepackage[`*драйвер*`]{graphicx}`

   где *драйвер* — это название вашего конвертера DVI в PostScript. Самый широко используемый называется `dvips`. Название драйвера требуется потому, что не существует стандарта включения графики в TeX. Зная название *драйвера*, graphicx может выбрать правильный способ вставить информацию о графике в `.dvi` файл так, чтобы драйвер ее понял и смог корректно вставить `.eps` файл.

3. Для включения *файла* в ваш документ пользуйтесь командой

   `\includegraphics[`*ключ*=*значение*, ... `]{`*файл*`}`

   Необязательный параметр принимает вид списка разделенных запятыми пар *ключей* и соответствующих *значений*. *Ключи* можно использовать для изменения ширины, высоты, поворота включаемой графики. Таблица 4.1 перечисляет самые важные ключи.

Таблица 4.1: Названия ключей для пакета graphicx

`width`	масштабирует графику до указанной ширины
`height`	масштабирует графику до указанной высоты
`angle`	поворачивает графику против часовой стрелки
`scale`	масштабирует графику

Возможно, пример сделает это понятнее:

```
\begin{figure}
\begin{center}
\includegraphics[angle=90, width=0.5\textwidth]{test.eps}
\end{center}
\end{figure}
```

Здесь включается графика, записанная в файл `test.eps`. Она *сначала* поворачивается на 90 градусов и *затем* масштабируется до конечной ширины в 0.5 ширины стандартного абзаца. Пропорции сохраняются, поскольку не указано конкретной высоты. Параметры высоты и ширины могут также быть указаны в абсолютных размерах. Обратитесь к таблице 5.5 на странице 67. Если вы хотите знать об этом больше, прочтите [8] и [11].

больше одной страницы. Некоторые принтерные драйверы нужно в явном виде настраивать для генерации EPS.

4.2. Библиография

Окружение `thebibliography` генерирует библиографию. Каждый элемент начинается с

`\bibitem{`*маркер*`}`

Затем *маркер* используется для дальнейших ссылок на книгу, статью или труд.

`\cite{`*маркер*`}`

Элементы библиографии нумеруются автоматически. Параметр после команды `\begin{thebibliography}` устанавливает максимальную ширину этих номеров. В следующем примере `{99}` указывает LaTeX, что ни один из номеров элементов не будет шире, чем число '99'.

Partl [1] предложил, чтобы …

Библиография

[1] H. Partl: *German TeX*, TUGboat Vol. 9, No. 1 ('88)

```
Partl~\cite{pa} предложил,
чтобы \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Vol.~9, No.~1 ('88)
\end{thebibliography}
```

Для использования в бо́льших проектах обратите внимание на программу BibTeX. Она входит в большинство поставок TeX. BibTeX позволяет вам поддерживать библиографическую базу данных и извлекать из нее ссылки, имеющие отношение к тому, что вы цитировали в вашем труде. Визуальное представление библиографий, генерируемых BibTeX, основано на концепции стилей, что позволяет вам создавать библиографии, следуя любому из существующих стилей библиографии.

4.3. Указатели

Полезной особенностью многих книг является их предметный указатель. Указатели могут создаваться автоматически при помощи LaTeX и сопровождающей программы `makeindex`[1]. В этом введении будут изложены только базовые команды генерации указателя. Более глубокое изложение смотрите в *The LaTeX Companion* [3].

[1] На системах, не поддерживающих длинные имена файлов, программа может называться `makeidx`.

Таблица 4.2: Примеры синтаксиса ключей указателя

Пример	**Вид указателя**	**Комментарий**
`\index{hello}`	hello, 1	Обычный элемент
`\index{hello!Peter}`	Peter, 3	Подчиненный ‘hello’ элемент
`\index{Sam@\textsl{Sam}}`	*Sam*, 2	Форматированный ключ
`\index{Lin@\textbf{Lin}}`	**Lin**, 7	То же
`\index{Jenny|textbf}`	Jenny, **3**	Форматированная страница
`\index{Joe|textit}`	Joe, *5*	То же

Для включения возможностей LaTeX в преамбуле должен загружаться пакет makeidx:

```
\usepackage{makeidx}
```

а специальные команды должны быть разрешены помещением в преамбулу команды

```
\makeindex
```

Содержимое указателя создается командами

```
\index{ключ}
```

где *ключ* является элементом указателя. Вы вводите команды указателя в том месте текста, куда этот элемент должен указывать. Таблица 4.2 объясняет синтаксис аргумента *ключ* несколькими примерами.

По мере обработки входного файла LaTeX, каждая команда `\index` записывает соответствующий элемент указателя вместе с номером текущей страницы в специальный файл. Файл имеет то же имя, что и входной файл LaTeX, но другое расширение имени (`.idx`). Если теперь повторно обработать входной файл, этот отсортированный указатель включается в документ в то место, где LaTeX находит команду

```
\printindex
```

Пакет showidx, входящий в LaTeX 2_ε, печатает все элементы указателя на левом поле текста. Это весьма полезно при проверке текста и сверке указателя.

4.4. Настраиваемые колонтитулы

Пакет fancyhdr[1], написанный Piet van Oostrum, предоставляет несколько простых команд, позволяющих вам настраивать верхние и нижние колонтитулы документа. Если вы сейчас взглянете на верх этой страницы, то увидите одно их возможных применений этого пакета.

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
% этим мы убеждаемся, что заголовки глав и
% разделов используют нижний регистр.
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}  % убираем текущие установки для колонтитулов
\fancyhead[LE,RO]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % оставляем место для линейки
\fancypagestyle{plain}{%
   \fancyhead{} % на обычных страницах убираем колонтитулы
   \renewcommand{\headrulewidth}{0pt} % и линейку
}
```

Рис. 4.1: Пример настройки fancyhdr

Сложность в настройке колонтитулов в том, чтобы включить туда вещи вроде заголовков раздела или главы. LaTeX достигает этого в два этапа. В определениях колонтитулов можно использовать команды `\rightmark` и `\leftmark`, представляющих заголовки текущей главы и раздела, соответственно. Значения этих двух команд меняются при обработке команд `\chapter` или `\section`.

Для большей гибкости команды `\chapter` и ей подобные не переопределяют `\rightmark` и `\leftmark` сами, а вызывают еще одну команду, называющуюся `\chaptermark`, `\sectionmark` или `\subsectionmark`, ответственную за переопределение `\rightmark` и `\leftmark`.

Так что, если вы хотите изменить вид названия главы в верхнем колонтитуле, вы просто переопределяете команду `\chaptermark`.

Рисунок 4.1 показывает, как можно настроить пакет fancyhdr так, чтобы колонтитулы выглядели почти так, как они выглядят у этого буклета. В любом случае

[1] Доступный из `CTAN:macros/latex/contrib/supported/fancyhdr`

вам рекомендуется ознакомиться с документацией к пакету по адресу, упомянутому в сноске.

4.5. Пакет verbatim

Ранее в этом введении вы познакомились с *окружением* `verbatim`. В этом разделе вы узнаете про *пакет* verbatim. Пакет verbatim представляет из себя повторную реализацию окружения `verbatim` с исправлением некоторых его ограничений. Само по себе это не замечательно[1], но, кроме того, в него была добавлена некоторая функциональность, и вот почему пакет здесь упоминается. Пакет verbatim предоставляет команду

`\verbatiminput{`*файл*`}`

которая позволяет вам включать текстовый файл в ваш документ, как если бы его содержимое находилось внутри окружения `verbatim`.

Так как пакет verbatim входит в комплект 'tools', вы найдете его установленным на большинстве систем. Если вы хотите узнать о нем больше, обязательно прочитайте [9].

[1]Это для американцев не замечательно. Для нас интересно то, что пакет verbatim включает команду `\verbatim"@font`, позволяющую использовать произвольный шрифт, например, русский, что невозможно в окружении `verbatim` без модификации стандартного поведения LaTeX. — **прим. переводчика**

5 Настройка LaTeX

Произведенные с использованием изученных до сих пор команд документы будут выглядеть вполне приемлемо для широкой аудитории. Они не выглядят очень модно, подчиняясь зато всем правилам хорошего тона в наборе, поэтому их легко читать и на них приятно смотреть.

Но бывают ситуации, в которых LaTeX не предоставляет команду или окружение, удовлетворяющие вашим ожиданиям, или производимый некоторой существующей командой вывод не отвечает вашим требованиям.

В этой главе даются некоторые советы по обучению LaTeX новым вещам, и тому, как сделать его вывод отличающимся от того, что производится по умолчанию.

5.1. Новые команды, окружения и пакеты

Как вы заметили, все вводимые в этой книге команды верстаются в рамке и включаются в указатель в конце книги. Вместо того, чтобы напрямую использовать необходимые для этого команды LaTeX, автор создал пакет, в котором определил новые команды и окружения для этой цели. Теперь можно просто писать:

\dum

```
\begin{command}
\ci{dum}
\end{command}
```

В этом примере используются как новое окружение, называющееся `command` и отвечающее за рисование рамки вокруг команды, так и новая команда, называющаяся `\ci` и верстающая название команды и заносящая соответствующий элемент в указатель. Вы можете в этом убедиться, поискав команду `\dum` в указателе в конце книги, где вы найдете запись для `\dum`, указывающую на эту страницу.

Если автор когда-нибудь решит, что ему не нравятся больше команды, сверстанные в рамке, он просто изменит определение окружения `command`. Это намного проще, чем пройти по всему документу, выискивая все места, где использованы общие команды LaTeX для рисования рамки вокруг слов.

5.1.1. Новые команды

Чтобы добавить ваши собственные команды, пользуйтесь командой

`\newcommand{`*название*`}[`*число*`]{`*определение*`}`

Обычно эта команда требует двух аргументов. *Название* команды, которую вы создаете, и *определение* команды. Аргумент *число* в квадратных скобках не обязателен. Он применяется для создания новых команды, которые, в свою очередь, принимают до 9 аргументов.

Следующие два примера должны вам помочь получить представление о команде. Первый пример определяет новую команду, называющуюся `\tnss`, что является сокращением от “The Not So Short Introduction to LaTeX 2$_\varepsilon$”. Такая команда пригодится, если вам много раз приходится писать название этой книги.

“The not so Short Introduction to LaTeX 2$_\varepsilon$” … “The not so Short Introduction to LaTeX 2$_\varepsilon$”

```
\newcommand{\tnss}{The not
   so Short Introduction to
   \LaTeXe}
% в теле документа:
``\tnss'' \ldots{} ``\tnss''
```

Следующий пример иллюстрирует использование аргумента *число*. Метка `#1` заменяется на заданный аргумент. Если вы хотите использовать более одного аргумента, пользуйтесь `#2`, и так далее.

- The *Не очень* краткое введение в LaTeX 2$_\varepsilon$
- The *Очень* краткое введение в LaTeX 2$_\varepsilon$

```
\newcommand{\txsit}[1]
    {The \emph{#1} краткое
      введение в \LaTeXe}
% в теле документа
\begin{itemize}
\item \txsit{Не очень}
\item \txsit{Очень}
\end{itemize}
```

LaTeX не позволит вам создать новую команду, которая бы изменяла уже существующую. Но для случая, когда вы явно хотите изменить существующую команду, есть специальная команда: `\renewcommand`. Она имеет тот же синтаксис, что и команда `\newcommand`.

В некоторых случаях может пригодиться команда `\providecommand`. Она работает так же, как `\newcommand`, но, если команда уже определена, то LaTeX 2$_\varepsilon$ ее молча проигнорирует.

5.1.2. Новые окружения

Аналогично команде `\newcommand`, существует команда для создания вашего собственного окружения. Команда `\newenvironment` имеет следующий синтаксис:

`\newenvironment{`*название*`}[`*номер*`]{`*начало*`}{`*конец*`}`

Подобно команде `\newcommand`, `\newenvironment` можно использовать с необязательным аргументом, или без него. Материал, заключенный в аргумент *начало*, обрабатывается до обработки текста внутри окружения. Материал, заключенный в аргумент *конец*, обрабатывается, когда встречается команда `\end{`*название*`}`. Следующий пример иллюстрирует использование команды `\newenvironment`.

■ Мои смиренные подданные … ■

```
\newenvironment{king}
 {\rule{1ex}{1ex}%
      \hspace{\stretch{1}}}
 {\hspace{\stretch{1}}%
      \rule{1ex}{1ex}}

\begin{king}
Мои смиренные подданные\ldots
\end{king}
```

Аргумент *номер* используют так же, как и для команды `\newcommand`. LaTeX контролирует, чтобы вы не определяли уже существующее окружение. Если вы заходите все же это сделать, пользуйтесь командой `\renewenvironment`. Она имеет тот же синтаксис, что ли `\newenvironment`.

Команды, использованные в этом примере, будут разъяснены позже: описание команды `\rule` см. на стр. 72, команда `\stretch` описана на стр. 66, а описание команды `\hspace` находится на стр. 66.

5.1.3. Ваш собственный пакет

Когда вы определяете множество новых окружений и команд, преамбулы ваших документов становятся очень большими. В этой ситуации представляется разумным создать пакет LaTeX, содержащий определения всех ваших команд и окружений. Потом можно командой `\usepackage` использовать пакет в ваших документах.

Создание пакета в основном состоит из переноса содержимого вашей преамбулы в отдельный файл с именем, заканчивающимся на `.sty`. Есть только одна специальная команда, которую вы должны использовать

`\ProvidesPackage{`*название пакета*`}`

в самом начале файла с вашим пакетом. `\ProvidesPackage` указывает LaTeX название пакета, что позволяет ему выдавать осмысленное сообщение об ошибке, когда вы

```
% Пакет для демонстрации. Tobias Oetiker.
\ProvidesPackage{demopack}
\newcommand{\tnss}{Не очень краткое введение в \LaTeXe}
\newcommand{\txsit}[1]{\emph{#1} краткое
                       введение в \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

Рис. 5.1: Пример пакета

пытаетесь включать пакет дважды. Иллюстрация 5.1 показывает маленький пример пакета, содержащего определенные в вышеприведенных примерах команды.

5.2. Шрифты и их размеры

5.2.1. Команды смены шрифта

LaTeX выбирает подходящее начертание и размер шрифта, основываясь на логической структуре документа (разделы, сноски, …). Иногда может быть желательно сменить шрифт вручную. Для этого вы можете пользоваться командами, перечисленными в таблицах 5.1 и 5.2. Действительный размер каждого шрифта определяется дизайном и зависит от класса и опций документа. Таблица 5.3 показывает абсолютные размеры, соответствующие этим командам в стандартных классах документов.

Маленький, **полужирный**, большой, *курсив*.

```
{\small Маленький,
  \textbf{полужирный},
  \Large большой,
  \textit{курсив}.}
```

Важная особенность LaTeX 2_ε заключается в том, что атрибуты шрифта независимы. Это значит, что вы можете давать команды смены размера или даже семейства шрифта, сохраняя при этом установки атрибутов наклона или насыщенности.

В *математическом режиме* вы можете использовать *команды* смены шрифта, чтобы временно выйти из *математического режима* и ввести нормальный текст. Если вы хотите переключиться на другой шрифт для верстки математики, то для этого существует отдельный набор команд. Смотрите таблицу 5.4.

В связи с командами смены размера шрифта заметную роль играют фигурные скобки. Они используются для построения *групп*. Группы ограничивают область действия большинства команд LaTeX.

Ему нравятся большие и маленькие буквы.

```
Ему нравятся {\LARGE
  большие и {\small
    маленькие} буквы}.
```

Таблица 5.1: Шрифты

`\textrm{...}`	прямой шрифт	`\textsf{...}`	без засечек
`\texttt{...}`	`пишущая машинка`		
`\textmd{...}`	нормальный	`\textbf{...}`	**полужирный**
`\textup{...}`	прямой шрифт	`\textit{...}`	*курсив*
`\textsl{...}`	*наклонный шрифт*	`\textsc{...}`	КАПИТЕЛЬ
`\emph{...}`	*выделенный шрифт*	`\textnormal{...}`	обычный

Таблица 5.2: Размеры шрифта

`\tiny`	крошечный	`\Large`	еще больше
`\scriptsize`	очень маленький	`\LARGE`	очень большой
`\footnotesize`	довольно маленький	`\huge`	огромный
`\small`	маленький	`\Huge`	громадный
`\normalsize`	нормальный		
`\large`	большой		

Таблица 5.3: Абсолютные размеры шрифтов в стандартных классах

Размер	*10pt (по умолчанию)*	*опция 11pt*	*опция 12pt*
`\tiny`	5pt	6pt	6pt
`\scriptsize`	7pt	8pt	8pt
`\footnotesize`	8pt	9pt	10pt
`\small`	9pt	10pt	11pt
`\normalsize`	10pt	11pt	12pt
`\large`	12pt	12pt	14pt
`\Large`	14pt	14pt	17pt
`\LARGE`	17pt	17pt	20pt
`\huge`	20pt	20pt	25pt
`\Huge`	25pt	25pt	25pt

Команды, влияющие на размер шрифта, влияют также на расстояние между строками, но только если соответствующий абзац заканчивается внутри области действия команды. Поэтому закрывающая фигурная скобка } не должна стоять слишком рано. Заметьте положение команды `\par` в следующих двух примерах:

Не читайте это! Это неправда. Верьте мне!

```
{\Large Не читайте это! Это
неправда. Верьте мне!\par}
```

Это тоже неправда. Но помните, что я вру.

```
{\Large Это тоже неправда.
Но помните, что я вру.}\par
```

Если вы хотите применить команду изменения размера к целому абзацу текста или больше того, то для этого лучше использовать синтаксис окружения.

Это неправда. Но что в наши дни . . .

```
\begin{Large}
  Это неправда. Но
  что в наши дни\ldots
\end{Large}
```

Это избавит вас от подсчета множества фигурных скобок.

5.2.2. Опасность!

Как отмечено в начале этой главы, опасно усеивать ваши документы явными командами, вроде только что описанных, потому что это противоречит основной идее LaTeX: разделению логической и визуальной разметки вашего документа. Это значит, что, если вы пользуетесь одними и теми же командами смены шрифта в разных местах для верстки специального вида информации, вы должны использовать `\newcommand` и определить команду, «оборачивающую» в себя команду смены шрифта.

Таблица 5.4: Математические шрифты

Команда	*Пример*	*Вывод*
`\mathcal{...}`	`$\mathcal{B}=c$`	$\mathcal{B}=c$
`\mathrm{...}`	`$\mathrm{K}_2$`	K_2
`\mathbf{...}`	`$\sum x=\mathbf{v}$`	$\sum x=\mathbf{v}$
`\mathsf{...}`	`$\mathsf{G\times R}$`	$\mathsf{G\times R}$
`\mathtt{...}`	`$\mathtt{L}(b,c)$`	$\mathtt{L}(b,c)$
`\mathnormal{...}`	`$\mathnormal{R_{19}}\neq R_{19}$`	$\mathnormal{R_{19}}\neq R_{19}$
`\mathit{...}`	`$\mathit{ffi}\neq ffi$`	$\mathit{ffi}\neq ffi$

Не **входите** в эту комнату. Она занята **машиной** неизвестного назначения.

```
% в преамбуле или пакете
\newcommand{\danger}[1]{\textbf{#1}}
% в документе
Не \danger{входите} в эту комнату.
Она занята \danger{машиной}
неизвестного назначения.
```

Этот подход имеет то преимущество, что вы позже можете решить, что хотите использовать другое визуальное представление опасности[1], нежели `\textbf`, без необходимости пробираться через весь документ, отыскивая все вхождения `\textbf` и определяя, отмечает ли каждое из них опасность или что-нибудь другое.

5.2.3. Совет

Для завершения нашего путешествия в мир шрифтов и их размеров, позвольте дать вам один совет:

> **Помните!** *Чем* бОльШЕ шрифтов ВЫ используете **в** вашем документе, *тем* легче ЕГО ЧИТАТЬ и тем ***красивее*** он буДеТ.

5.3. Интервалы

5.3.1. Интервалы между строками

Если вам нужны бо́льшие интервалы между строками, то их значение можно изменить помещением в преамбулу команды

`\linespread{`*коэффициент*`}`

Для печати «через полтора интервала» пользуйтесь `\linespread{1.3}`, для печати «через два интервала» — `\linespread{1.6}`. По умолчанию этот коэффициент равен 1.

5.3.2. Форматирование абзацев

Два параметра в LaTeX влияют на верстку абзацев. Поместив в преамбулу определения вида

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

вы измените внешний вид абзацев. Эти две строчки увеличивают расстояние между абзацами и устанавливают абзацный отступ равным нулю. В Европе абзацы часто отделяют пробелами и не делают в них отступа. Однако, имейте в виду, что это

[1] danger — **прим. переводчика**

влияет также и на оглавление: его строки тоже становятся теперь более разреженными. Чтобы этого избежать, эти команды можно перенести из преамбулы документа куда-нибудь после `\tableofcontents`, или не использовать их совсем, потому что в профессиональной книжной верстке используется выделение абзацев красной строкой, а не пробелами.

Если вы хотите сделать абзацный отступ в не имеющем его абзаце, то вставьте в начало абзаца команду[1]

```
\indent
```

Понятно, что эффект от нее будет только если `\parindent` не установлен равным нулю.

Для создания абзаца без отступа первой командой абзаца можно сделать

```
\noindent
```

Это может быть удобно, когда вы начинаете документ с текста, а не с команды секционирования.

5.3.3. Горизонтальные интервалы

LaTeX автоматически определяет пробелы между словами и предложениями. Чтобы добавить горизонтальный пробел, пользуйтесь

`\hspace{`*длина*`}`

Если такой интервал должен быть выдержан, даже если он приходится на начало или конец строки, используйте `\hspace*`, а не `\hspace`. В простейшем случае *длина* — это просто число и единица измерения. Наиболее важные единицы перечислены в таблице 5.5.

Тут пробел в 1,5см.

```
Тут\hspace{1.5cm}пробел
в 1,5см.
```

Команда

`\stretch{`n`}`

генерирует специальный «резиновый» пробел. Он растягивается, заполняя все оставшееся места на строке. Если на одной строке встречаются две команды `\hspace{\stretch{`n`}}`, то они растягиваются пропорционально своим коэффициентам.

x x x

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

[1] Для добавления отступа к первому абзацу после каждого заголовка раздела пользуйтесь пакетом indentfirst из комплекта 'tools'.

Таблица 5.5: Единицы размерности в TeX

`mm`	миллиметр ≈ 1/25 дюйма
`cm`	сантиметр = 10 mm
`in`	inch = 25.4 mm
`pt`	пункт ≈ 1/72 дюйма ≈ $\frac{1}{3}$ mm
`em`	примерная ширина буквы 'M' текущего шрифта
`ex`	примерная высота буквы 'x' текущего шрифта

5.3.4. Вертикальные интервалы

Интервалы между абзацами, разделами, подразделами, ... определяются LaTeX автоматически. При необходимости дополнительный пробел *между двумя абзацами* можно добавить командой

`\vspace{`*длина*`}`

Обычно эта команда вставляется между двумя пустыми строчками. Если это пространство должно сохраняться вверху или внизу страницы, используйте вариант команды со звездочкой: `\vspace*`.

Команду `\stretch` вместе с `\pagebreak` можно применять для верстки текста на последней строке страницы или для вертикального центрирования текста на странице.

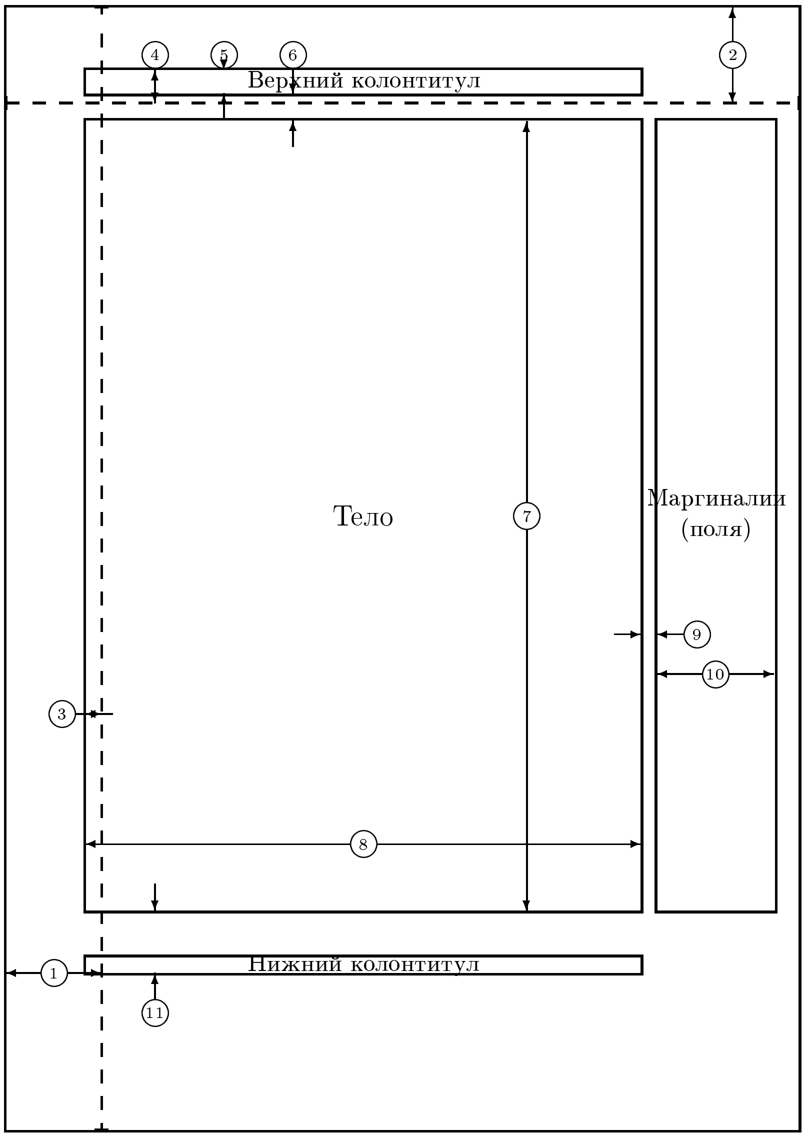
```
Некий текст\ldots

\vspace{\stretch{1}}
А это окажется на последней строке страницы.\pagebreak
```

Дополнительный пробел между двумя строками *одного* абзаца или внутри таблицы указывается командой

`\\[`*длина*`]`

5.4. Компоновка страницы

LaTeX 2_ε позволяет указать размер бумаги в команде `\documentclass`. Затем он автоматически выбирает правое поле. Но иногда предопределенные значения могут вас не устроить. Безусловно, вы их можете изменить. Иллюстрация 5.2 показывает



1	`1 дюйм + \hoffset`	2	`1 дюйм + \voffset`
3	`\oddsidemargin = -12pt`	4	`\topmargin = -25pt`
5	`\headheight = 18pt`	6	`\headsep = 20pt`
7	`\textheight = 595pt`	8	`\textwidth = 418pt`
9	`\marginparsep = 12pt`	10	`\marginparwidth = 89pt`
11	`\footskip = 47pt`		`\marginparpush = 6pt (не показано)`
	`\hoffset = 0pt`		`\voffset = 0pt`
	`\paperwidth = 597pt`		`\paperheight = 845pt`

Рис. 5.2: Параметры компоновки страницы

все параметры, которые можно изменить. Она была сгенерирована пакетом layout из комплекта 'tools'[1].

ПОДОЖДИТЕ! ... прежде, чем немедленно броситься делать эту слишком узкую страницу слегка пошире, потратьте несколько секунд на размышления. Подобно другим вещам, выбор компоновки страницы в LaTeX весьма продуман.

Безусловно, если сравнить со страницей, выданной свежеустановленным MS Word, то страницы LaTeX выглядят ужасно узкими. Однако, взгляните на вашу любимую книгу[2] и посчитайте количество букв на одной строчке. Вы обнаружите, что на каждой строчке не больше 66 букв. Теперь повторите это со страницей LaTeX. Вы увидите, что и здесь тоже около 66 букв в строке. Опыт показывает, что при большем количестве букв чтение затрудняется, потому, что глазам становится труднее переходить от конца одной строки к началу следующей. Именно поэтому газеты часто верстаются в несколько колонок.

Так что, увеличивая ширину вашего текста, имейте в виду, что вы затрудняете жизнь его читателям. Однако, достаточно предупреждений, вам был обещан рассказ о том, как же это сделать ...

LaTeX предоставляет две команды для изменения этих параметров. Их обычно используют в преамбуле документа.

Первая команда присваивает фиксированное значение любому параметру:

\setlength{*параметр*}{*длина*}

Вторая команда прибавляет длину к любому параметру:

\addtolength{*параметр*}{*длина*}

Она даже более полезна, чем `\setlength`, потому что позволяет вам делать настройку относительно существующих установок. Чтобы добавить сантиметр к общей ширине текста, например, в преамбулу нужно поместить следующее:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

Здесь вам может быть интересен пакет calc, который позволяет использовать арифметические операции в аргументе `\setlength` и в других местах, где аргументами являются численные значения.

5.5. Еще о длинах

Всегда, когда это возможно, избегайте использовать в документах абсолютных величин. Лучше основывайтесь на ширине или высоте других элементов страницы.

[1] CTAN:macros/latex/packages/tools

[2] Имея в виду настоящую печатную книгу, выпущенную уважаемым издательством.

Для ширины иллюстрации этим может служить `\textwidth`, чтобы она заполняла страницу целиком.

Следующие три команды позволяют определить ширину, высоту и глубину текстовой строки.

```
\settoheight{команда}{текст}
\settodepth{команда}{текст}
\settowidth{команда}{текст}
```

Нижеследующий пример показывает возможное применение этих команд.

$$a^2 + b^2 = c^2$$

Где: a, b – прилегают к прямому углу прямоугольного треугольника.

c – одинокая гипотенуза этого треугольника.

d – вообще тут не участвует. Вот загадка …

```
\flushleft
\newenvironment{vardesc}[1]{%
  \settowidth{\parindent}{#1:\ }
  \makebox[0pt][r]{#1:\ }}{}

\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}

\begin{vardesc}{Где}$a$,
$b$ -- прилегают к прямому углу
прямоугольного треугольника.

$c$ -- одинокая гипотенуза
этого треугольника.

$d$ -- вообще тут не участвует.
Вот загадка\ldots
\end{vardesc}
```

5.6. Блоки

LaTeX выстраивает страницы, передвигая блоки. Сначала каждая буква является маленьким блоком, который приклеивается к другим буквам, формируя слово. Слова склеиваются с другими словами, но специальным эластичным клеем, который может растягиваться или сжиматься, так, чтобы в точности заполнить строку.

Надо признать, что это довольно упрощенная версия того, что происходит на самом деле, но идея в том, что TeX всегда работает с блоками и клеем. Не только буква может быть блоком. Вы можете поместить в блок практически все, что угодно, не исключая и другие блоки. Каждый блок затем обрабатывается LaTeX, как если бы это была отдельная буква.

В предыдущих главах вы уже встречали некоторые блоки, хотя об этом и не говорилось. Примерами могут быть окружение `tabular` или `\includegraphics`, оба производящие блок. Это значит, что вы легко можете сверстать рядом две таблицы или иллюстрации. Только убедитесь, что их общая ширина не превышает `\textwidth`.

Вы также можете упаковать любой абзац в блок или командой

`\parbox[`*поз*`]{`*ширина*`}{`*текст*`}`

или окружением

`\begin{minipage}[`*поз*`]{`*ширина*`}` текст `\end{minipage}`

Параметр *поз* может принимать одну из букв `c, t` или `b`, контролируя вертикальное выравнивание блока по отношению к базовой линии окружающего текста. *Ширина* принимает аргументом длину, определяющую ширину блока. Основное отличие между `\minipage` и `\parbox` — в том, что внутри `\parbox` можно использовать не все команды и окружения, тогда как внутри `\minipage` можно практически все.

В то время, как `\parbox` упаковывает целый абзац, разбивая строчки и прочее, существует класс блоковых команд, работающих только на горизонтально расположенном материале. Одну из них мы уже знаем. Она называется `\mbox` и просто упаковывает последовательность блоков, что можно использовать для предотвращения переноса LaTeX двух слов. Так как вы можете помещать одни блоки в другие, эти упаковщики горизонтальных блоков чрезвычайно гибки.

`\makebox[`*ширина*`][`*поз*`]{`*текст*`}`

Ширина определяет ширину результирующего блока так, как его видно снаружи[1]. Кроме выражений длины, вы тут можете использовать `\width`, `\height`, `\depth` и `\totalheight`. Они устанавливаются равными значениям, полученным измерением параметров *текста*.[2] Параметр *поз* принимает однобуквенное значение: **c**: центрировать, **l**: отжать влево, **r**: отжать вправо или **s**: равномерно заполнить блок текстом.

Команда `\framebox` работает в точности так же, как `\makebox`, но рисует рамку вокруг текста.

Следующий пример показывает некоторые возможности использования команд `\makebox` и `\framebox`.

[1]Это означает, что она может быть меньше, чем материал внутри блока. В предельном случае вы можете даже установить ее в 0pt, так что текст внутри блока верстается, вообще не оказывая влияния на окружающие блоки.

[2]Ширина, высота, глубина и общая высота (высота плюс глубина) текста, соответственно — **прим. переводчика**.

центр

р а с т я н у т ы й

Я теперь в рамке!

Ой, я слишком толстый

ничего, я тоже Можете это прочитать?

```
\makebox[\textwidth]{%
    ц е н т р}\par
\makebox[\textwidth][s]{%
    р а с т я н у т ы й}\par
\framebox[1.1\width]{Я теперь
  в рамке!} \par
\framebox[0.8\width][r]{Ой,
    я слишком толстый} \par
\framebox[1cm][l]{ничего,
  я тоже}
Можете это прочитать?
```

Теперь, когда мы управляем горизонталью, очевидный следующий шаг — вертикаль. Никаких проблем. Команда

`\raisebox{`*сдвиг*`}[`*глубина*`][`*высота*`]{`*текст*`}`

позволяет вам определить вертикальные характеристики блока. В первых трех параметров можно использовать `\width`, `\height`, `\depth` и `\totalwidth`, чтобы получить размеры аргумента *текст*.

Aaaaaaaa кричал он, но даже стоящий рядом не заметил, что с ней случилось что-то ужасное.

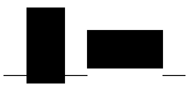
```
\raisebox{0pt}[0pt][0pt]{\Large%
\textbf{Aaaa\raisebox{-0.3ex}{a}%
\raisebox{-0.7ex}{aa}%
\raisebox{-1.2ex}{a}%
\raisebox{-2.2ex}{a}%
\raisebox{-4.5ex}{a}}}
кричал он, но даже стоящий рядом
не заметил, что с ней случилось
что-то ужасное.
```

5.7. Линейки и распорки

Несколько страниц назад вы могли отметить команду

`\rule[`*сдвиг*`]{`*ширина*`}{`*высота*`}`

При обычном использовании она генерирует простой черный блок.



```
\rule{3mm}{.1pt}%
\rule[-1mm]{5mm}{1cm}%
\rule{3mm}{.1pt}%
\rule[1mm]{1cm}{5mm}%
\rule{3mm}{.1pt}
```

Это можно использовать для рисования вертикальных и горизонтальных линий. Например, линия на титульном листе нарисована командой `\rule`.

Специальным случаем является линейка, у которой нет ширины, но есть определенная высота. В профессиональной верстке ее называют *распоркой*. Ее используют, чтобы обеспечить определенную минимальную высоту элемента страницы. Вы можете использовать ее, чтобы сделать строку окружения `tabular` имеющей определенную минимальную высоту.

Pittprop …
Strut

```
\begin{tabular}{|c|}
\hline
\rule{0pt}{4ex}Pittprop \ldots\\
\hline
\rule{0pt}{4ex}Strut\\
\hline
\end{tabular}
```

Библиография

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, второе издание, 1994, ISBN 0-201-52983-1.

[2] Donald E. Knuth. *The TeXbook,* Volume A of *Computers and Typesetting*, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9.

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.

[4] Каждая установка LaTeX должна содержать так называемый *LaTeX Local Guide*, объясняющий особенности локальной системы. Он должен находиться в файле, называющемся `local.tex`. К сожалению, некоторые ленивые администраторы такого документа не предоставляют. В таком случае просите о помощи местного LaTeX гуру.

[5] LaTeX3 Project Team. *LaTeX 2$_\varepsilon$ for authors.* Включен в поставку LaTeX 2$_\varepsilon$ как `usrguide.tex`.

[6] LaTeX3 Project Team. *LaTeX 2$_\varepsilon$ for Class and Package writers.* Включен в поставку LaTeX 2$_\varepsilon$ как `clsguide.tex`.

[7] LaTeX3 Project Team. *LaTeX 2$_\varepsilon$ Font selection.* Включен в поставку LaTeX 2$_\varepsilon$ как `fntguide.tex`.

[8] D. P. Carlisle. *Packages in the 'graphics' bundle.* Входит в состав комплекта 'graphics' как `grfguide.tex`, доступен оттуда же, откуда ваша поставка LaTeX.

[9] Rainer Schöpf, Bernd Raichle, Chris Rowley. *A New Implementation of LaTeX's verbatim Environments.* Входит в состав комплекта 'tools' как `verbatim.dtx`, доступен оттуда же, откуда ваша поставка LaTeX.

[10] Graham Williams. *The TeX Catalogue* полный список множества пакетов, имеющих отношение к TeX и LaTeX Доступен в Интернет по адресу `CTAN:help/Catalogue/catalogue.html`

[11] Keith Reckdahl. *Using EPS Graphics in LaTeX 2$_\varepsilon$ Documents* объясняет все, что вы когда бы то ни было хотели знать про EPS файлы и их использование в документах LaTeX. Доступен в Интернет по адресу `CTAN:info/epslatex.ps`

Предметный указатель