Содержание

1 Введение

Первая часть этого раздела содержит краткий обзор философии и истории LaTeX. Вторая часть — фокусируется на основных элементах структуры документов LaTeX. После чтения этого раздела вы должны иметь общее представление о том, как работает LaTeX.

1.1 Названия

1.1.1 TeX

В начале 70–х годов XX века американский математик Дональд Кнут (Donald E. Knuth), готовя к изданию второй том книги «Искусство программирования» и просматривая пробные оттиски страниц, пришёл к выводу, что качество отпечатанного текста было очень низким. Однако, будучи серьёзным учёным — математиком и программистом, — он обратился за помощью в ту сферу науки, где и был профессионалом, а именно к цифровому представлению текста, который в простейшем виде является размещением нулей и единиц (т. е., выражаясь более образно, «есть чернила» — «нет чернил») на шаблоне листа.

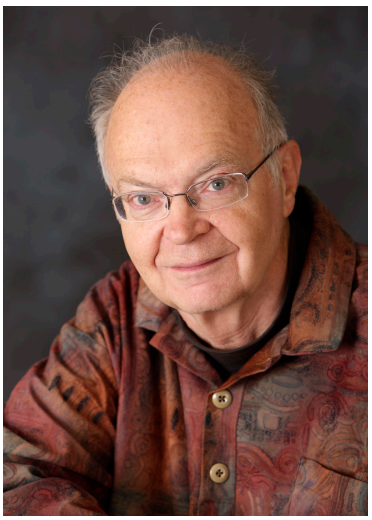

Рис. 1: Дональд Эрвин Кнут

Поэтому Кнут начал прорабатывать и анализировать «традиционно» лучшие по типографским меркам правила набора текстов и способы создания новых шрифтов, поскольку подходящих для него шрифтов не существовало.

Кнут начал писать TeX в 1977 году для изучения потенциала цифрового печатающего оборудования, которое начало появляться в это время, надеясь, в особенности, обратить тенденцию ухудшения типографского качества, которую он видел на примере его собственных книг и статей. Первая версия TeX вышла в 1978 году, а в том виде, в каком мы его сегодня используем, был выпущен в 1982 году с некоторыми добавлениями в 1989 и 2008 годах (лучшая поддержка 8-битных символов и различных языков). TeX знаменит своей чрезвычайной стабильностью, работой на различных типах компьютеров и практи-

чески полным отсутствием ошибок. Номер версии TEX сходится к π и сейчас равен 3.1415926.

Кнут полагал, что вся работа займёт не более полугода, но, как оказалось, он ошибся: создание новой системы заняло около 10 лет.

В эти годы вместе с ним работала целая команда математиков и программистов, получавшая поддержку Американского математического общества.

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

1.1.2 LATEX

LATEX — наиболее популярный макропакет, позволяющий авторам верстать и печатать их работы с высоким типографским качеством, при помощи заранее определённых, профессиональных макетов. LATEX был написан Leslie Lamport. В качестве механизма для верстки он использует TEX. Сейчас LATEX поддерживает Frank Mittelbach.

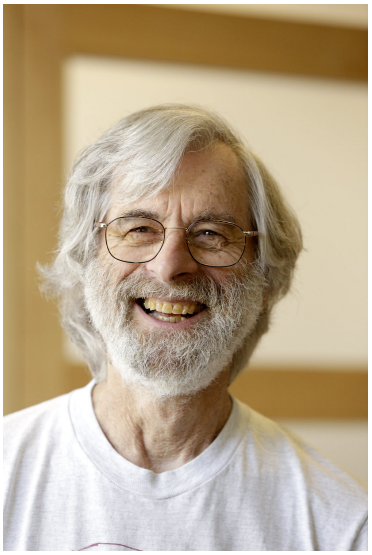

Рис. 2: Лесли Лэмпорт

Важно заметить, что ни один из макропакетов для TEX'а не может расширить возможностей TEX (всё, что можно сделать в LATEX'е, можно сделать и в PlainTEX'е), но, благодаря различным упрощениям, использование макропакетов зачастую позволяет избежать весьма изощрённого программирования.

LATEX позволяет автоматизировать многие задачи набора текста и подготовки статей, включая набор текста на нескольких языках, нумерацию разделов и формул, перекрёстные ссылки, размещение иллюстраций и таблиц на странице, ведение библиографии и др. Кроме базового набора существует множество пакетов расширения LATEX. Первая версия была выпущена Лесли Лэмпортом в 1984 году; текущая версия, LATEX 2_ε, после создания в 1994 году испытывала некоторый период нестабильности, окончившийся к концу 1990-х годов, а в настоящее время стабилизировалась (хотя раз в год выходит новая версия).

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

При работе в LATEX, автор сначала должен подготовить с помощью любого текстового редактора файл с текстом, содержащий команды для LATEX'а.

Такие файлы по традиции имеют расширение `tex`. Потом нужно обработать исходный файл с помощью программы-транслятора (компилятора).

В современных версиях LaTeX'а в результате компиляции сразу получается pdf-файл (в настоящее время, как правило, компиляция выполняется программой pdfLaTeX, XeLaTeX или LuaLaTeX)

После этого полученный pdf-файл можно распечатать на принтере, посмотреть на экране (текст будет в таком же виде, как он появится на печати) и т. д. Если в документ нужно внести изменения, автор редактирует исходный файл — и повторяет компиляцию исходного файла.

1.2 Основы

1.2.1 Автор, дизайнер и верстальщик

Для того, чтобы опубликовать рукопись, авторы отдают ее 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 отдаёт ее верстальщику, который верстает книгу в соответствии с этими инструкциями.

Дизайнер 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 издательской среде LaTeX, сам LaTeX берет на себя роль дизайнера книги, используя TeX в качестве верстальщика. Но LaTeX — это всего лишь программа, и, следовательно, нуждается в более чётких 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 подхода WYSIWYG[1], принятого в большинстве современных текстовых процессоров и издательских систем, таких как *Microsoft Word* или *Adobe InDesign*. 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.

[1] What you see is what you get.

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

1.2.2 Дизайн макета

Типографский дизайн — это профессия. Неопытные авторы часто допускают серьёзные 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 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. 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;
- LaTeX поощряет авторов писать хорошо структурированные документы, так как именно так LaTeX и работает — путём 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:

- хотя предопределённые макеты имеют множество настраиваемых параметров, создание полностью нового макета документа не очень просто и занимает много времени;
- сложно писать неструктурированные и неорганизованные документы.

1.3 Исходный файл

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 введёте 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 ещё один, так как эта команда (\\) используется для разрыва строки. Вместо этого пользуйтесь командами `$\backslash$` (для математического режима) и `\testbackslash` (для текстового режима). Обе эти команды дают на печати символ «\».

Коротко расскажем о смысле некоторых спецсимволов. Символ % является символом начала комментария — все символы, расположенные в строке после него (и сам %), TeX игнорирует.

Фигурные скобки ограничивают группы в исходном файле. Знак доллара ограничивает математические формулы. При наборе математических формул, также, используются знаки _ и ^ («знак подчёркивания» и «крышка»). Знак тильда «~» обозначает «неразрывной пробел» между словами.

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 небуквенного символа.

LaTeX игнорирует пробелы после команд. Если вы хотите получить пробел после команды, вы должны поместить «{}» 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 \TeX{}ников и \TeX пертов.\\
Сегодня "--- \today
```

Я слышал, что Кнут разделяет людей, работающих с TeX на TeXников и TeXпертов.
Сегодня — 9 сентября 2016 г.

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 \textsl{положиться}
на меня!
```

Вы можете *положиться* на меня!

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

Пожалуйста, начните новую строчку прямо тут!
Спасибо!

1.3.4 Звёздочка после имени команды

В LaTeX'е некоторые команды и окружения имеют варианты, в которых непосредственно после имени команды или окружения ставится звёздочка «*». Например, команда `\section` означает «начать новый раздел документа», а команда `\section*` означает «начать новый раздел документа, не нумеруя его и не добавлять в оглавление».

1.3.5 Комментарии

Когда в процессе обработки входного файла LaTeX встречает символ %, он игнорирует остаток текущей строки, возврат каретки и все пробелы в начале следующей строки.

Этим можно пользоваться для добавления в исходный файл замечаний, которые не будут выводиться на печать.

```
Это Spercal% здесь можно добавить
         ifragilist%  комментарий
     icexpialidocious
```

Это Spercalifragilisticexpialidocious

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

Для более длинных комментариев можно также пользоваться окружением comment, предоставляемым пакетом verbatim. Это означает, что, для использования окружения comment, вы должны к преамбуле вашего документа добавить команду \usepackage{verbatim}:

```
Это "--- еще один
\begin{comment}
довольно глупый,
но полезный
\end{comment}
пример вставки комментариев
в ваш документ.
```

Это — еще один пример вставки комментариев в ваш документ.

Заметьте, что это не будет работать внутри сложных окружений, например, математики.

1.3.6 Группы

Важным понятием TeX'а является понятие группы. Чтобы понять, что это такое, рассмотрим пример.

При обработке TeX'ом исходного файла набор текста в каждый момент идет каким-то вполне определенным шрифтом (он называется текущим шрифтом). Изначально текущим шрифтом является «обычный» прямой шрифт (по-английски *«roman»*). Команда \slshape переключает текущий шрифт на наклонный, а \upshape выполняет обратное переключение. Аналогичным образом команды \bfseries и \mdseries меняют жирность шрифта.

```
Полужирный шрифт начнется
с \bfseries этого слова.
Снова \mdseries светлый,
теперь \slshape наклонный,
до нового переключения;
вновь \upshape прямой.
```

Полужирный шрифт начнется с **этого слова. Снова** светлый, теперь *наклонный, до нового переключения; вновь* прямой.

В этом примере можно обойтись и без команд `\mdseries` и `\upshape` (отменяющих действие предыдущих команд). Для этого часть текста, которую вы хотите оформить полужирным или наклонным шрифтом, можно заключить в фигурные скобки, и дать команду `\bfseries` или `\slshape` внутри этих скобок. Тогда сразу же после закрывающей фигурной скобки TEX «забудет» про то, что шрифт переключался, и будет продолжать набор тем шрифтом, который был до скобок:

```
Полужирным шрифтом набрано
только {\bfseries это
слово}; после скобок все
идет, как прежде.
```

Полужирным шрифтом набрано только **это слово**; после скобок все идет, как прежде.

Сами по себе фигурные скобки не создают никакого текста и не влияют на шрифт; единственное, что они делают — это ограничивают *группу* внутри файла.

Как правило, задаваемые командами TEX'а изменения различных параметров (в нашем случае — текущего шрифта) действуют в пределах той группы, внутри которой была дана соответствующая команда; по окончании группы (после закрывающей фигурной скобки, соответствующей той фигурной скобке, что открывала группу) все эти изменения забываются и восстанавливается тот режим, который был до начала группы.

Некоторые команды, называемые *глобальными*, сохраняют своё действие и за пределами той группы, где они были употреблены. Всякий раз, когда идёт речь о *глобальной команде*, это будет специально оговариваться. Впрочем, глобальных команд в LATEX'е мало.

1.3.7 Структура исходного файла

LaTeX-файл всегда должен начинаться с команды

```
\documentclass[options]{class}
```

которая задаёт класс (стиль оформления) документа.

Классы — это простые текстовые файлы с расширением `cls`, которые содержат определённые правила оформления документа: нумерация разделов, оформление колонтитулов, наличие и оформление титульного листа и т. д.

Например,

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

Слово `article` в фигурных скобках указывает, что документ будет оформлен, как статья. В квадратных скобках указаны необязательные аргументы, которые указывают LaTeX набирать документ шрифтом размером 12 пунктов на бумаге формата A4.

Класс `article` — это самый популярный класс документа, который подходит для оформления большинства документов.

Кроме класса `article` существует ещё несколько стандартных классов, которые имеются в любом минимальном варианте издательской системы LaTeX: `book` (для оформления книг), `report` (технический отчет), `letter` (для оформления деловых писем так, как это принято в США), `proc` (для оформления изданий типа «труды конференции»), `slides` (для слайдов презентаций). Сокращённый каталог доступных классов и стилей в LaTeX, а также их краткое описание можно посмотреть здесь.

Следующей обязательной командой является —

```
\begin{document}
```

Текст до команды `\begin{document}` называется преамбулой.

Преамбула обычно содержит команды, производящие дополнительную настройку выбранного класса печатного документа, подключения дополнительных пакетов, а также определения новых команд LaTeX. Если вы поместите обычный текст до команды `\begin{document}`, то LaTeX, скорей всего, выдаст сообщение об ошибке.

Собственно сам текст документа начинается после `\begin{document}`, и заканчивается командой —

```
\end{document}
```

Всё, что следует за командой `\end{document}`, LaTeX игнорирует.

Перечисленные три команды дают основные указания компилятору, как должен выглядеть печатный документ. Если одна из них пропущена, LaTeX выдаст сообщение об ошибке при компиляции входного файла.

Ниже приведён самый минимальный, составленный по всем правилам, LaTeX-файл

```
\documentclass[a4paper,12pt]{article}
\begin{document}  % Конец преамбулы, начало текста.
   Hello world!
\end{document}    % Конец текста.
```

Рис. 3: Минимальный файл LaTeX

Если вы до сих пор не работали с системой LaTeX, сейчас самое время попробовать выполнить компиляцию такого документа на локальном компьютерре или на одном из сайтов с онлайн-редактором LaTeX-документов.

Все примеры LaTeX-документов, приведённые в этом пособие, выложены на сайте ShareLaTeX, где можно бесплатно зарегистрироваться по ссылке.

Если документ содержит русский текст надо обязательно добавить в преамбулу ещё несколько команд (рис. 4).

1.3.8 Параметры классов документов

В таблице 1 перечислены часто используемые опции стандартных классов документов.

```
\documentclass[a4paper,12pt]{article}
% ============================================================
\usepackage{cmap}  % поиск и копирование в PDF документах
\usepackage[T2A]{fontenc}   % внутренняя кодировка шрифта
\usepackage[utf8]{inputenc}  % кодировка исходного текста
\usepackage[english,russian]{babel} % локализация и переносы
\begin{document} % Конец преамбулы, начало текста.
   Hello world!

   Привет, мир!
\end{document}   % Конец текста.
```

Рис. 4: Минимальный файл LaTeX с поддержкой русского языка

Таблица 1: 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`.
`leqno`	Формулы нумеруются слева, а не справа.
`onecolumn`, `twocolumn`	Заставляет LaTeX набирать документ в один столбец или 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.

1.3.9 Окружения

Ещё одна важная конструкция LaTeX'а — это окружение (environment).

Окружение — это фрагмент файла, начинающийся с текста

```
\begin{имя_окружения}
```

где `имя_окружения` представляет собой первый обязательный (и, возможно, не единственный) аргумент команды `\begin`. Заканчивается окружение командой

```
\end{имя_окружения}
```

(команда `\end` имеет только один аргумент — имя завершаемого ею окружения). Например:

```
Все строки этого абзаца будут
центрированы; переносов не будет,
если только какое-то слово,
как в дезоксирибонуклеиновой
кислоте, не длинней строки.
```

Все строки этого абзаца будут центрированы; переносов не будет, если только какое-то слово, как в дезоксирибонуклеиновой кислоте, не длинней строки.

Каждой команде `\begin`, открывающей окружение, должна соответствовать закрывающая его команда `\end` (с тем же именем окружения в качестве аргумента).

Важным свойством окружений является то, что они действуют и как фигурные скобки: часть файла, находящаяся внутри окружения, образует группу. Например, внутри окружения `center` в вышеприведённом примере можно было бы сменить шрифт, скажем, командой `\itshape`, и при этом после команды `\end{center}` восстановился бы тот шрифт, что был перед окружением.

1.4 Единицы длины

1.4.1 Фиксированные длины

Многие параметры, используемые LaTeX'ом, являются размерами.

В табл. 2 собраны единицы длины, которые можно использовать в TeX'е при задании размеров.

Примечание: TeX'овский пункт является единицей измерения, принятой в англо-американской типометрии; он отличается от пункта, принятого в континентальной Европе (в том числе и в России). Единица измерения, называемая

в TEX'е пунктом Дидо, соответствует пункту, к которому привыкли отечественные полиграфисты.

Десятичные числа в TEX'е и LATEX'е могут быть записаны любым способом — английским (через десятичную точку) или европейским (через десятичную запятую).

Следует отметить, **что задавать размеры нужно только с указанием единицы измерения**, даже если длина имеет нулевое значение, например, чтобы задать нулевую длину необходимо написать `0pt` или `0cm`.

Таблица 2: 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
`bp`	большой пункт (1 дюйм = 72 bp)
`pc`	пика (1 pc = 12 pt)
`dd`	пункт Дидо (1157 dd = 1238 pt)
`cc`	цицеро (1cc = 12dd)
`em`	примерная ширина буквы 'M' текущего шрифта
`ex`	примерная высота буквы 'x' текущего шрифта

Значения присваиваются параметру длины с помощью команды

```
\setlength{\имя параметра}{значение длины}
```

Например, величина абзацного отступа задаётся параметром `\parindent`, который обычно устанавливается в зависимости от размера текущего шрифта. Для того, чтобы задать абзацный отступ равный пяти знакам[2], нужно в преамбуле документа подключить пакет `calc` и задать параметру `\parindent` расстояние равное пяти знакам «x»:

```
\usepackage{calc}
\setlength{\parindent}{\widthof{xxxxx}}
```

Пакет `calc`, позволяет использовать арифметические операции в аргументе `\setlength` и в других местах, где аргументами являются численные значения.

[2] ГОСТ Р 7.0.11-2011. 5.3.7 …Абзацный отступ должен быть одинаковым по всему тексту и равен пяти знакам

Команда `\widthof{xxxxx}` вычисляет горизонтальное расстояние, которое занимают пять символов «x», включая расстояние между символами.

После добавления этих команд в преамбулу, LaTeX будет всегда автоматически вычислять величину абзацного отступа в зависимости от размера текущего шрифта.

1.4.2 Переменные длины

Некоторым параметрам можно задавать переменную (*«резиновую»*) длину. Это расстояния, которые могут увеличиваться или уменьшаться.

Например, для того чтобы LaTeX мог растягивать или сжимать вертикальные расстояния между абзацами, нужно задать команду —

```
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex},
```

которая создаст дополнительный интервал между абзацами, величиной `1ex`, но который может быть увеличен до `1.5ex` или уменьшен до `0.8ex`

Случай растяжимых промежутков настолько важен, что в LaTeX'е для него предусмотрена специальная команда `\fill`, которая задаёт промежуток нулевого размера, но обладающий способностью бесконечно растягиваться.

Например, если написать

```
\clearpage\vspace*{\fill}
\begin{center}
Заголовок
\end{center}
\vspace*{\fill}\clearpage
```

то слово «заголовок» будет расположено точно по центру отдельной страницы, созданной командами `\clearpage`. Перед командой `\fill` в аргументе `\vspace` или `\vspace*` можно поставить коэффициент — целое число или десятичную дробь, и тогда растяжимость умножится на этот коэффициент.

1.5 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 3.

Таблица 3: 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*`}`

1.6 Встречающиеся типы файлов

Работая с LaTeX, вы вскоре начнете путаться в куче файлов с различными расширениями. Ниже перечислены различные типы файлов, используемые при работает с TeX. Заметьте, что это не полный список расширений, но, если вы найдете не упомянутое расширение, которое считаете важным, — уведомите, пожалуйста, автора.

Следующие файлы генерируются, когда LaTeX обрабатывает входной файл:

`.dvi` Device Independent file (файл, не зависящий от устройства). Это — основной результат запуска LaTeX. Содержимое его можно увидеть при помощи

программы отображения DVI, или распечатать программой `dvips` или аналогичной.

`.log` Содержит детальный отчет о том, что происходило в последний прогон компиляции.

`.toc` Хранит заголовки всех разделов. Читается в следующий проход компиляции и используется при генерации оглавления.

`.lof` Аналог `.toc` для списка иллюстраций.

`.lot` То же, для списка таблиц.

`.aux` Еще один файл, передающий информацию между проходами компиляции. Кроме все прочего, используется для генерации перекрестных ссылок.

`.idx` Если ваш документ содержит предметный указатель, LaTeX помещает все слова для указателя в этот файл. Обработайте его программой `makeindex`. Подробнее смотрите в разделе **??** на странице **??**.

`.ind` Обработанный файл `.idx`, готовый ко включению в ваш документ при следующем проходе компиляции.

`.ilg` Журнал работы `makeindex`.

`.pdf` Portable Document Format (PDF) — межплатформенный формат электронных документов, разработанный фирмой Adobe Systems. В первую очередь предназначен для представления полиграфической продукции в электронном виде. Для просмотра существует множество программ, а также официальная бесплатная программа Adobe Reader. Значительное количество современного профессионального печатного оборудования имеет аппаратную поддержку формата PDF, что позволяет производить печать документов в данном формате без использования какого-либо программного обеспечения. Это — основной результат запуска pdfLaTeX.