Представляемое справочное руководство является частью документации по системе TEX — популярной в научных кругах системы компьютерного набора, предназначенной для издания книг высокого полиграфического качества и особенно книг, содержащих много математических формул.

Работа по созданию и сопровождению русифицированной версии этой системы выполняется при поддержке гранта РФФИ 96–7–89406 “Представление научной текстографической документаци в компьютерных сетях России на базе системы TEX”. Все создаваемые в рамках этого проекта документы по мере готовности помещаются на общедоступный сервер protex.ihep.su, где вместе с русифицированной системой TEX–LATEX и сопутствующим математическим обеспечением уже находится следующая документация:

- *Лисина М.В.* Plain TEX. Основные понятия и каталог команд.
- *Клименко С.В., Грицаенко И.А.* LaTeX, руководство пользователя.
- *Клименко С.В., Лисина М.В.* LaTeX и его команды. Основные понятия и каталог команд.
- *Клименко С.В., Лисина М.В., Фомина Н.М.* AMSTEX — руководство пользователя.

Кроме того, на сервере находятся гипертекстовые справочники по командам TEXа и LATEXа.

Тем не менее в настоящее время не все пользователи TEXа имеют устойчивый доступ к сетевым ресурсам, а многие просто все еще предпочитают традиционные “бумажные” издания, которые можно почитать в спокойной домашней обстановке. Поэтому мы считаем целесообразным продолжать традицию и издавать руководства в виде препринтов ИФВЭ, что возможно благодаря поддержке и пониманию дирекции Института.

Мы чрезвычайно благодарны Е.В. Панкратьеву за плодотворную кооперацию в работе над совместным проектом РФФИ 96–7–89406, С.Н. Соколову и В.Н. Ноздрачеву за ценные замечания и дополнения, возникшие в результате многочисленных обсуждений, В.И. Лисину за моральную и техническую поддержку, Л.С. Сиколенко за замечательный рисунок для обложки, а также С.А. Лесенко за его высокопрофессиональную помощь при подготовке оригинал-макета.

* * *

Настоящее справочное руководство относится к LATEXу версии 2.09. Оно не является ни полным и всесторонним описанием LATEXа, ни учебником для начинающих — для этой цели есть другие книги. В частности, можно посоветовать недавно вышедший в ИФВЭ препринт И.А. Грицаенко и С.В. Клименко “LATEX Руководство для пользователей” [4].

Здесь же мы использовали принцип расположения материала, который ранее опробовали в своей работе над каталогом команд plain TEXа [5], а именно, принцип словарей, когда в начале приводится краткая грамматика языка, а уже затем в алфавитном порядке объясняются слова. Так и наше руководство состоит из двух частей: первой (вспомогательной) с кратким неформальным изложением “грамматики” LATEXа и второй (основной) с полным каталогом команд и ключевых слов LATEXа, включая некоторые команды plain TEXа, которые “понимает” LATEX и которые необходимы для понимания приводимых примеров (эта часть имеет значительное пересечение с вышедшим три года назад “Каталогом Plain TEXа” [5]).

Описания команд расположены в алфавитном порядке и сопровождаются примерами и иллюстрациями, показавшимися нам интересными. Эта (вторая) часть руководства представляет интерес в качестве справочника как для начинающих, так и для опытных пользователей, которым совсем не обязательно читать первую часть.

Содержание

I Основные понятия LaTeXa **5**

1 Введение **5**

2 Команды **5**
2.1 Специальные символы 5
2.2 Командные последовательности 6
2.3 Длины и параметры длины 6
2.3.1 Длины 6
2.3.2 Параметры длины 7
2.4 Командные скобки 8
2.5 Декларации 8
2.6 Счетчики и текущие `\ref`-значения 9
2.7 Подвижные аргументы и хрупкие команды 10

3 Общая структура входного файла **10**
3.1 Параметры стиля 11
3.2 Предложения, абзацы 11
3.3 Знаки препинания 12

4 Выделение информации **12**
4.1 Центрирование информации и сдвиг ее 12
4.2 Списки 13
4.3 Боксы 13
4.4 Таблицы 14

5 Акценты **14**

6 Плавающие вставки **14**

7 Диаграммы и картинки **15**

8 Моды **15**
8.1 Абзацная мода 15
8.2 Математическая мода 16
8.3 LR мода 17

9 Стили страниц **17**
9.1 Параметры стиля 18
9.2 Изменение размера выходной страницы 19
9.3 Подготовка текста в две колонки 19

10 Определение собственных команд **20**

11 Математические символы **21**
11.1 Ординарные символы 21
11.1.1 Строчные греческие буквы 21
11.1.2 Прописные греческие буквы 22
11.1.3 Прописные греческие курсивные буквы 22
11.1.4 Прописные каллиграфические буквы 22
11.1.5 Дополнительные символы 23

11.2 Символы больших операторов . . . 23
11.2.1 Символы элементарных функций . . . 23
11.2.2 Определение собственных больших операторов . . . 24
11.3 Символы бинарных операторов . . . 24
11.3.1 Определение собственных символов операторов . . . 25
11.4 Символы бинарных отношений . . . 25
11.4.1 Стрелки . . . 26
11.5 Ограничители . . . 26
11.5.1 Открывающие символы . . . 26
11.5.2 Закрывающие символы . . . 26
11.5.3 Определение собственных ограничителей . . . 27
11.6 Символы пунктуации . . . 27
11.7 Составные конструкции . . . 27
11.7.1 Акценты . . . 27
11.7.2 Подчеркивание, надчеркивание и связки . . . 28
11.7.3 Нижние и верхние индексы . . . 28
11.7.4 Дроби и подобные им структуры . . . 28
11.8 Многострочные конструкции . . . 28
11.8.1 Командные скобки `array` . . . 28
11.8.2 Матрицы в TEXe . . . 29
11.8.3 Массивы уравнений и помеченные формулы . . . 29

12 Выбор шрифта **29**
12.1 Изменение стиля шрифта . . . 29
12.2 Изменение размера шрифта . . . 30
12.3 Загрузка шрифтов . . . 30
12.4 Шрифты в математической моде . . . 30

13 Библиографии и алфавитные указатели **32**
13.1 Командные скобки `thebibliography` . . . 32
13.2 Использование BIBTEXa . . . 33
13.2.1 Структура bib-файла . . . 34
13.2.2 Подготовка библиографии . . . 36
13.3 Получение алфавитного указателя . . . 38

14 Подготовка “глоссария” **38**

15 Использование команд Plain TEXa **38**
15.1 Команды табулирования . . . 39
15.2 Вывод, сноски и рисунки . . . 39
15.3 Команды выбора шрифта . . . 39
15.4 Выравнивание уравнений . . . 40
15.5 Разное . . . 40

16 Работа над ошибками **40**

II Каталог команд LATEXa **42**

Библиография **159**

Список рисунков

1 Расположение заголовка, тела и основания страницы. 17
2 Использование командных скобок `thebibliography`. 32
3 Необязательный аргумент команды `\bibitem`. 33
4 Различные виды типов публикаций *publication-type*. 35
5 Статус полей элемента базы данных BibTeX для различных типов публикаций. 37
6 Формула Раманужана и команды, которыми она была получена. 48
7 “Короткие” и “длинные” выключенные формулы. 52
8 Хвост мыши из “Алисы в стране чудес” — команды для получения. 53
9 Хвост мыши из “Алисы в стране чудес”. 54
10 Ограничители в TeXe (примеры кодировки). 56
11 Ограничители в TeXe (результаты). 57
12 Команды рубрикации стиля документа `article`. 59
13 Пример использования стиля документа `article`. 60
14 Декларации, изменяющие шрифт. 63
15 Команды рубрикации стилей документа `book` и `report`. 65
16 Полный набор окружностей и кругов для построения рисунков в командных скобках `picture`. 72
17 Пример использования команд `\makebox`, `\dashbox` и `\framebox` в командных скобках `picture`. 76
18 Шесть fill-команд. 82
19 Размеры шрифтов. 90
20 Действие аргумента *pos* в команде `\framebox`. 91
21 Пример использования командных скобок `letter`. 103
22 Пример письма, отформатированного LaTeXом. 104
23 Наклоны прямых линий, допустимые в командных скобках `picture`. 106
24 Параметры командных скобок `list`. 108
25 Пример использования макрокоманды `\multiput`. 117
26 Пример использования команды `\oval`. 122
27 Результаты игр в биллиард между четырьмя лучшими игроками в классификационных соревнованиях сезона 1990–1991 годов. 139
28 Параметры, которые действуют на вид страницы результата. 148
29 Наклоны векторов, допустимые в командных скобках `picture`. 154

Часть I
Основные понятия LaTeXa

1 Введение

Общий процесс получения документа с использованием LaTeXa сильно отличается от стандартных текстовых процессоров. LaTeX определенно не является WYSIWYG системой[1]. Вы начинаете работу с создания текста с помощью любого текстового редактора. Кроме символов конца строки (и символа конца файла, вставляемого операционной системой), файл будет содержать только видимые символы, включая символ пробела:[2]

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
А Б В Г Д Е Ж З И К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ъ Ы Ь Э Ю Я
а б в г д е ж з и к л м н о п р с т у ф х ц ч ш щ ъ ы ь э ю я
0 1 2 3 4 5 6 7 8 9
! " # $ % & ’ ( ) * + , - . / :
; < = > ? @ [ \ ] ^ _ ‘ { | } ~
```

В большинстве операционных систем имена файлов состоят из двух частей — *основного* имени и *расширения*. Части имени разделяются точкой.

После того, как файл создан, его надо обработать LaTeXом. Способ, каким это будет сделано, сильно зависит от используемой операционной системы, но во многих системах надо выполнить команду операционной системы `latex` *file* или `latex` *file*`.tex`. Входные файлы LaTeXa обычно имеют расширение `tex`, и если Вы укажите только основное имя входного файла, некоторые операционные системы будут предполагать именно это расширение. На Вашем терминале появятся различные сообщения, и если предположить, что LaTeX при обработке файла не обнаружит ошибок, Вы увидите, что по окончании работы он создал три дополнительных файла, а именно, *file*`.aux`, *file*`.dvi` и *file*`.log`. Файл с расширением `aux` (auxiliary — вспомогательный) содержит информацию для перекрестных ссылок, а файл с расширением `log` — все, что появилось на терминале во время работы LaTeXa, а также дополнительную информацию. Файл с расширением `dvi` (device independent — машинно-независимый) наиболее важен. Для получения документа надо пропустить dvi-файл через еще одну программу. Например, если принтер понимает язык PostScript, Вам надо выполнить программу, которая преобразует dvi-файл в PostScript-файл. Такая программа часто называется `dvi2ps`, `dvitops` или `dvips`. Как только Вы получите PostScript-файл, его можно послать на принтер, который выдаст окончательный результат.

2 Команды

2.1 Специальные символы

Следующие десять символов TeX считает специальными:

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

Символ `%`, например, сигнализирует, что все, что следует за ним в строке, где он находится, является комментарием. Все, что расположено между символом `%` и ближайшим символом

[1] WYSIWYG — это акроним для “what you see is what you get” (“что вы видите, то и получаете”), иногда это записывается так: “what you see is all you’ve got”.

[2] Во всей книге шрифт пишущей машинки использован для указания ввода LaTeXa, но иногда — в редких случаях — он используется и для указания на команды операционной системы или на внешние имена файлов.

конца строки (включая сам этот символ конца строки) TEXом игнорируется и в выходном документе, полученном LATEXом, не появляется. Фигурные скобки { и } используются для целей группирования и всегда должны быть парными. Символ \ — это escape-символ (или командный символ).[3]

Значение всех специальных символов объяснено в каталоге. (Если Вы хотите включить в выходной документ один из этих специальных символов (за исключением бэкслэша, тильды и шляпки), поставьте перед ним бэкслэш. Так, для получения & надо ввести `\&`, а для $ — `\$`. Чтобы получить бэкслэш, надо ввести `\backslash`, но только в математической моде).

2.2 Командные последовательности

Каждая команда LATEXа начинается с командного символа, за которым следует либо одна или несколько букв, либо один неалфавитный символ. Такие команды называются в TEXе *командными последовательностями*. Командные последовательности, состоящие из командного символа и одной или нескольких букв, называются *командными словами*, а состоящие из командного символа и одного неалфавитного символа — *командными символами*. Например, `\noindent`, `\it` и `\c` — это командные слова, а `\/` — это командный символ. Заметим, что TEX различает прописные и строчные буквы, так что, например `\eg` и `\Eg` — это два разных командных слова. Также заметим, что в командных словах могут встречаться только буквы, но не цифры. Командные слова оканчиваются либо пробелом, либо любым символом, не являющимся буквой. Если командное слово заканчивается пробелом, то этот пробел игнорируется. Кроме того, TEX трактует последовательность любого числа пробелов как один пробел.

Около 300 командных последовательностей называются *примитивами* — это элементарные операции самого низкого уровня, которые не разлагаются на более простые функции. Все остальные определены, в конечном счете, в терминах примитивов, т.е. являются *макрокомандами*. Так, формат PLAIN состоит из более 600 макрокоманд, большая часть которых используется и в LATEXе.

Команды могут иметь аргументы. Обычно в LATEXе обязательные аргументы команд заключаются в фигурные скобки, а необязательные — в квадратные; важные исключения из этого правила относятся к командам, встречающимся в командных скобках `picture`.

2.3 Длины и параметры длины

2.3.1 Длины

TEX оперирует следующими абсолютными единицами длины: `bp` (большой пункт), `cc` (цицеро), `cm` (сантиметр), `dd` (пункт Дидо), `in` (дюйм), `mm` (миллиметр), `pc` (пайка) и `sp` (масштабный пункт). Между ними выполняются следующие соотношения:

$$\begin{aligned}
1\text{ in} &= 2.54\text{ cm},\\
1\text{ in} &= 72.27\text{ pt},\\
1\text{ in} &= 72\text{ bp},\\
1\text{ cm} &= 10\text{ mm},\\
1\text{ pc} &= 12\text{ pt},\\
1\text{ pt} &= 65536\text{ sp}\\
1\text{ cc} &= 12\text{ dd},\\
1157\text{ dd} &= 1238\text{ pt}.
\end{aligned}$$

Масштабный пункт — это единица, в которой выражаются все хранящиеся в TEXе внутренние длины.

[3] Он не имеет ничего общего с клавишей escape на терминале компьютера.

В дополнение к этим абсолютным единицам длины, значения которых не зависят от текущего размера шрифта, имеются три относительные единицы, а именно, em, ex и mu (математическая единица), причем последняя из них разрешена только в математической моде. Традиционно em была равна ширине прописной буквы “M”, а ex — высоте строчной буквы “x”, но в шрифтах, которые использует TEX, это просто некие зависящие от шрифта единицы длины. В шрифтах гарнитуры Computer Modern, созданных Кнутом, ширина em-тире всегда равна одному em, ширина десятичных цифр равна половине em, высота обычных круглых скобок (и) — одному em, а высота буквы “x” — одному ex.

2.3.2 Параметры длины

Параметр длины — это команда длины, которая в LATEXе воздействует на внешний вид выходного результата. Параметры длины бывают *жесткими* и *эластичными*. Параметр эластичной длины — это параметр, реальное значение которого может растягиваться и сжиматься в зависимости от контекста, в котором он появляется, в то время как значение параметра жесткой длины фиксировано и никогда не растягивается и не сжимается. Например, значение параметра жесткой длины `\textheight` равно высоте тела страницы выходного результата, полученного LATEXом.

Параметры жесткой длины

Приведем список параметров жесткой длины LATEXа:

`\arraycolsep`	`\headsep`	`\marginparpush`
`\arrayrulewidth`	`\itemindent`	`\marginparsep`
`\bibindent`	`\labelsep`	`\marginparwidth`
`\columnsep`	`\labelwidth`	`\mathindent`
`\columnseprule`	`\leftmargin`	`\oddsidemargin`
`\doublerulesep`	`\leftmargini`	`\parindent`
`\evensidemargin`	`\leftmarginii`	`\rightmargin`
`\fboxrule`	`\leftmarginiii`	`\tabbingsep`
`\fboxsep`	`\leftmarginiv`	`\tabcolsep`
`\footheight`	`\leftmarginv`	`\textheight`
`\footnotesep`	`\leftmarginvi`	`\textwidht`
`\footskip`	`\linewidth`	`\topmargin`
`\headheight`	`\listparindent`	`\unitlength`

Чтобы изменить любой из них, надо включить во входной или в стилевой файл присваивание типа `\arraycolsep=5pt` или использовать команду `\setlength`, например так:

```
\setlength{\unitlength}{1mm}
```

или так:

```
\setlength{\rightmargin}{leftmargin}.
```

Параметры эластичной длины

Приведем список параметров эластичной длины LATEXа:

`\abovedisplayshotskip`	`\doubletextfloatsep`	`\partopsep`
`\abovedisplayskip`	`\floatsep`	`\textfloatsep`
`\baselineskip`	`\intextskip`	`\topsep`
`\belowdisplayshotskip`	`\itemsep`	`\topskip`
`\belowdisplayskip`	`\parsep`	
`\dblfloatsep`	`\parskip`	

Любой из этих параметров можно изменить присваиванием следующего вида:

```
\parskip=12pt plus 4pt minus 2pt
```

Это означает, что естественное значение эластичного параметра длины равно 12 пунктам — в TEXнологии оно называется *естественным пробелом* — но оно может растягиваться (или расширяться) на 4 пункта и сжиматься (или сокращаться) на 2 пункта. В присваиваниях значений параметрам эластичной длины *ключевые слова* `plus` и `minus` и связанные с ними длины являются необязательными; если какое-нибудь из них опущено, TEX считает его равным нулю.

2.4 Командные скобки

Некоторые комбинации команд в LATEXе называются *командными скобками*. Например, синтаксис командных скобок `document` такой:

`\begin{document}` *text* `\end{document}`

В этой книге при описании синтаксиса команд LATEXа слова и символы, напечатанные шрифтом пишущей машинки, представляют символы, которые именно так выглядят во входном файле, а слова, напечатанные курсивом, только обозначают во входном файле место, а пользователь сам решает, чему там быть.

Командные скобки `document` несколько необычны тем, что могут встретиться во входном файле только *один* раз; для других командных скобок является нормальной ситуация, когда они появляются во входном файле несколько раз и даже могут быть вложены в другие командные скобки. Имена командных скобок обычно полностью состоят из букв, хотя иногда они могут оканчиваться звездочкой. Имена командных скобок не могут содержать других символов, кроме букв и звездочки.

В LATEXе имеются 34 вида встроенных командных скобок, хотя с помощью деклараций `\newenvironment` и `\newtheorem` можно вводить новые командные скобки, а уже существующим можно дать новые значения с помощью декларации `\renewenvironment`. Приведем полный список предварительно определенных командных скобок:

`abstract`	`flushleft`	`table`
`array`	`flushright`	`table*`
`center`	`itemize`	`tabular`
`description`	`letter`	`tabular*`
`displaymath`	`list`	`thebibliography`
`document`	`math`	`theindex`
`enumerate`	`minipage`	`trivlist`
`eqnarray`	`picture`	`verbatim`
`eqnarray*`	`quotation`	`verbatim*`
`equation`	`quote`	`verse`
`figure`	`sloppypar`	
`figure*`	`tabbing`	

Командные скобки общего назначения `list` (и их сокращенная версия `trivlist`) используются для определения некоторых других командных скобок, а именно таких спископодобных командных скобок, как `center`, `description`, `enumerate`, `flushleft`, `flushright`, `itemize`, `quotation`, `quote`, `thebibliography` и `verse`.

2.5 Декларации

Декларация в LATEXе — это команда, которая не производит какого-либо результата — в выходном документе нет ни текста, ни символа, который прямо соответствовал бы декла-

рации из входного файла — но она некоторым образом проявляется в результате.[4] *Область действия* декларации `\it`, изменяющей тип шрифта, ограничена фигурными скобками.

2.6 Счетчики и текущие \ref-значения

В LaTeXe *счетчик* — это целочисленная переменная. Всего имеется 23 встроенных счетчика, причем с помощью команды `newcounter` можно определить и другие (иногда применение команды `newtheorem` также создает новый счетчик). Приведем полный список предварительно определенных в LaTeXe счетчиков:

`bottomnumber`	`figure`	`subparagraph`
`chapter`	`footnote`	`subsection`
`dbltopnumber`	`mpfootnote`	`subsubsection`
`enumi`	`page`	`table`
`enumii`	`paragraph`	`tocdepth`
`enumiii`	`part`	`topnumber`
`enumiv`	`secnumdepth`	`totalnumber`
`equation`	`section`	

Несколько этих счетчиков являются параметрами, которые действуют на расположение плавающих вставок, а именно счетчики `bottomnumber`, `dbltopnumber`, `topnumber` и `totalnumber`. Счетчик `tocdepth` управляет тем, что заносится в оглавление, а `secnumdepth` влияет на нумерацию единиц рубрикации.

Другие счетчики применяются для различной нумерации и эти номера, исключая номера сносок, могут использоваться для перекрестных ссылок. Значение счетчика можно вывести различными способами. Например, если *ctr* — это счетчик, то `\arabic`{*ctr*} выводит значение *ctr* арабскими цифрами, в то время как `\roman`{*ctr*} выводит это значение в виде строчного римского числа, `\Roman`{*ctr*} — в виде прописного римского числа, а `\alph`{*ctr*} и `\Alph`{*ctr*} выводят значение счетчика (но только если оно лежит в пределах между 1 и 26) соответственно строчными или прописными латинскими буквами[5]. Каждому счетчику *ctr* соответствует команда `\the`*ctr*, которая по умолчанию представляет собой текст, связанный с этой командой. Например, счетчик `page` содержит текущий номер страницы, но результирующий текст, который появляется на странице, хранится в команде `\thepage`, значение которой по умолчанию равно `\arabic{page}`. Но текстуальные результаты, связанные с предварительно определенными счетчиками, как показывает следующая таблица, не всегда имеют форму арабских цифр:

ctr	`\the`*ctr*
`footnote`	`\arabic{footnote}`
`mpfootnote`	`\alph{mpfootnote}`
`page`	`\arabic{page}`
`part`	`\Roman{part}`

Каждая команда `\label`{*key*} записывает в `aux`-файл команду такого вида:

`\newlabel`{*key*}{{*text*$_1$}{*text*$_2$}},

где *text*$_2$ — это номер той страницы, на которой появилась команда `\label`{*key*}, а *text*$_1$ зависит от контекста, в котором появилась команда `\label`{*key*}: внутри командных скобок

[4] Специалисты-компьютерщики — а может быть, также и другие — вероятно, должны быть смущены такой трактовкой понятия декларации, так как она включает присваивание. В информатике более принято считать, что декларации и команды, которые включают присваивания, это разные вещи.

[5] Если в команде `\documentstyle` задана опция `russian`, счетчик выводится русскими буквами.

`enumerate` — это метка, зависящая от глубины вложенности командных скобок, после команды `\caption` внутри командных скобок `figure`, `figure*`, `table` и `table*` — это метка рисунка или таблицы, внутри командных скобок `equation` и `eqnarray` — это метка текущей формулы, а во всех других случаях — метка, зависящая от команд рубрикации. Заметим, что в стилях документа `book` и `report` все эти метки начинаются с номера главы, за исключением меток, получаемых командными скобками `enumerate`.[6]

Когда во входной файл помещается команда `\pageref`{*key*}, то в выходной результат попадает текст $text_2$, а присутствие команды `\ref`{*key*} выводит текст $text_1$.

2.7 Подвижные аргументы и хрупкие команды

Когда входной файл обрабатывается LaTeXом, кроме `dvi`-файла записывается (или перезаписывается) еще несколько файлов и некоторые аргументы некоторых команд LaTeXа содержат текст, который записывается в эти дополнительные файлы. Например, если Ваш входной файл содержит команду `\makeindex` и какое-то количество команд `\index`{*text*}, то аргумент *text* попадет в `idx`-файл, который записывает LaTeX. Такой аргумент называется *подвижным* аргументом. (В некоторых случаях подвижный аргумент не приводит к записи информации в файл; если команда имеет подвижный аргумент, эта информация включается в глоссарий.)

Команды LaTeXа бывают либо *хрупкими*, либо *прочными*. Прочная команда — это такая команда, которая, если появляется в подвижном аргументе, передается в дополнительный файл точно в том же виде, в котором она встречается в этом аргументе. Однако при передаче хрупкой команды могут возникнуть проблемы, которые, правда, можно обойти, если перед хрупкой командой поставить команду `\protect`.

3 Общая структура входного файла

Каждый входной файл LaTeXа имеет следующую структуру:

```
\documentstyle[opt-list]{doc-style}
    dec-seq
\begin{document}
    text
\end{document}
```

Во входном файле перед командой `\documentstyle` может появиться только очень малое число команд, среди них `\batchmode`, `\errostopmode`, `\nonstopmode` и `\scrollmode`. Эти команды, задающие моду работы, объясняются в каталоге.

doc-style Основной стиль документа. Стандартные стили: `article`, `report`, `book` и `letter` (только для писем). Команда `\documentstyle` читает файл *doc-style*`.sty`.

opt-list Список из одной или нескольких опций стиля, разделяемых запятыми без пробелов. Стандартные опции LaTeXа: `11pt`, `12pt`, `twoside`, `twocolumn`, `titlepage`, `openbib`, `leqno`, `flegn`.

LaTeX реализует опции стилей для каждой указанной опции *op* следующим образом: если команда `\ds@`*op* определена (обычно основным стилем), то она выполняется, в противном случае считывается файл *op*. `sty`.

Та часть входного файла LaTeXа, которая располагается между командой `\documentstyle` и открывающей командной скобкой `document`, называется *преамбулой*, и она состоит из возможно пустой последовательности деклараций, которые влияют на окончательный вид входного документа. Сам текст заключается в командные скобки `document`.

[6] Отметим, что это некоторое упрощение того, что происходит на самом деле.

Заголовок создается с помощью команд `\title`, `\author` и `\date`, описывающих необходимую информацию, и команды `\maketitle`, генерирующей заголовок. Если авторов несколько, они разделяются командами `\and` в аргументе `\author`. Оглавление получается командой `\tableofcontents`, список таблиц — командой `listoftables`, а список иллюстраций — командой `listoffigures`

Если документ длинный, то его можно поделить на части: секции, подсекции и т.д. Единица рубрикации начинается одной из следующих команд рубрикации:

```
\part       \subsection       \paragraph
\chapter    \subsubsection    \subparagraph
\section
```

аргумент которой дает заголовок единицы и является его подвижным аргументом.

3.1 Параметры стиля

`\bibindent` Ширина дополнительного отступа последующих строк в блоке библиографии для опции стиля `openbib`.

`\columnsep` Ширина интервала между столбцами текста в стиле `twocolumn`.

`\columnseprule` Ширина вертикальной черты, помещаемой между столбцами текста в стиле `twocolumn`. По умолчанию эта величина равна нулю, что создает невидимую черту.

`\mathindent` Величина отступа формул от левого поля в опции стиля документа `fleqn`.

3.2 Предложения, абзацы

Предложения и абзацы вводятся в основном так, как и следует ожидать. TEX полностью игнорирует форматирование входного файла. Пустая строка указывает новый абзац (для этой цели можно также использовать команду plain TEXa `\par`). Точка, вопросительный или восклицательный знак считаются концом предложения, если они не следуют за прописной буквой. После них (а также после других знаков препинания) следует ставить пробелы, которые автоматически увеличиваются. Команда `\@` перед знаком пунктуации заставляет TEX трактовать его как конец предложения, а команда `\spac` после знака пунктуации дает пробел. Команда `~` дает пробел между словами, на котором TEX не начинает новую строку. Команда `\mbox` препятствует тому, чтобы TEX переносил аргумент на другую строку.

Абзацы начинаются с абзацного отступа, величина которого задается параметром `\parindent` и подавить который можно командой `\noindent`.

Можно явно задавать вертикальные и горизонтальные пробелы, используя как собственные команды LATEXa типа `\vspace` и `\hspace`, так и команды plain TEXa `\hskip`, `\vskip`, `\smallskip`, `\medskip`, `\bigskip`.

Логограммы TEX и LATEX создаются командами `\TeX` и `\LaTex`. Команда `\today` дает текущую дату, а `\ldots` — многоточие (...).

Текст выделяется с помощью декларации `\em`. Выделяемый текст обычно набирается курсивом. Команда `\/` должна идти сразу после буквы, набранной курсивом, если за ней следует прямой текст и если этот прямой текст не начинается с точки или с запятой.

Сноски печатаются с помощью команды `\footnote`, аргументом которой является текст сноски.

Математические формулы, содержащиеся в текстовой строке, заключаются в `\(` ... `\)` или `$`...`$`. Математические формулы, выделяемые на отдельной строке, заключаются в `\[` ... `\]` или `$$`...`$$`. Нижние и верхние индексы создаются с помощью команд `_` и `^`. Символ `'` дает знак прим (').

3.3 Знаки препинания

Тире

Наборщики используют четыре вида тире, а именно, em-тире (—), en-тире(–), дефис (-) и знак минус. Знак минус встречается только в математической моде.

Em-тире Em-тире получается тремя последовательными знаками минус в абзацной или LR моде. Em-тире часто используют парами, чтобы показать, что текст между ними является вводным.

En-тире En-тире получается из двух последовательных знаков минус в абзацной или LR моде. Оно используется для задания числовых диапазонов типа “стр.33–35” или “война 1939–1945”, между названиями местностей, как-то связанных между собой, например, “ось Рим–Берлин”, и между именами соавторов, чтобы не возникало путаницы с двойными фамилиями, например, “теорема Черча–Россера”. Также это тире используется вместо дефиса в составных прилагательных, как минимум одна из компонент которых состоит из составного слова или из двух слов. Например, “Нью-Йорк–Амстердамская компания” или “квази-общественная–квази-судебная организация”.

Дефис Тире, которое ставится в составных словах, например, “светло-голубой”. Получается одним знаком минус в абзацной или LR моде.

Знаки кавычек

Чтобы получить слово или фразу в одинарных кавычках, надо заключить его в апострофы или знаки одинарных кавычек. Чтобы получить слово или фразу в двойных кавычках, надо заключить его в пары апострофов или пары знаков кавычек.

Пробелы

TEX по умолчанию после точек, знака вопроса и восклицательного знака помещает пробел бóльшей величины, чем просто между словами. Исключением является случай, когда любой из этих знаков следует за заглавной буквой. Но если перед этими знаками стоит команда `\@`, то пробел после них будет также увеличиваться. Командная последовательность `\␣` вставляет обычный междусловный пробел. Заметим, что пробел после точки, восклицательного и вопросительного знаков оказывается больше, даже если за этими знаками следуют кавычки или скобки; в этих случаях указанный выше способ также хорошо работает.

4 Выделение информации

Короткие цитаты выделяются командными скобками `quote`, а длинные — `quotation`.

Командные скобки `verse` используются для стихов. Новая строфа начинается с пустой строки, а за строкой, которая не является концом строфы, следует команда `\\`; используйте `\\*` вместо `\\`, для того чтобы не допустить после строки перехода на другую страницу.

4.1 Центрирование информации и сдвиг ее

Центрирование информации производят командные скобки `center`. Командные скобки `flushleft` прижимают ее к левой границе страницы, а `flushright` — к правой. В каждых этих командных скобках для обозначения конца строки выходного результата можно использовать команду `\\`. Не следует путать эту команду конца строки с символом конца строки (клавиша return) во входном файле, который TEX трактует как обычный пробел.

Вместо этих командных скобок можно использовать декларации: вместо командных скобок `flushleft` — декларацию `\raggedright`, вместо командных скобок `flushright` — декларацию `\raggedleft`, а вместо командных скобок `center` — декларацию `\centering`. Здесь, однако, есть одно отличие: когда используются командные скобки, а не декларации, текст, расположенный внутри этих командных скобок, отделяется от предшествующего текста пробелом. Величина этого пробела равна либо сумме параметров `\topsep` и `\parskip` (если перед командными скобками нет ни пустой строки, ни команды `\par`), либо сумме трех параметров `\topsep`, `\parskip` и `\partopsep` (если перед командными скобками есть либо пустая строка, либо команда `\par`).[7]

4.2 Списки

Особый вид выделения — это списки. Список — это последовательность пунктов, набранных в абзацной моде с отступом от левого и правого полей. Каждый пункт начинается с метки. Метка может быть пустой, а отступ равен нулю, поэтому командные скобки, которые обычно не считаются списком, могут рассматриваться как список. Действительно, почти каждые командные скобки LaTeXa, которые начинают новую строку, определены как список. Делающими список командными скобками являются: `quote`, `quotation`, `verse`, `itemize`, `enumerate`, `description`, `thebibliography`, `center`, `flushright` и `flushleft`, а также теоремоподобные командные скобки, объявленные командой `\newtheorem`.

В LaTeXe предусмотрены два вида примитивных делающих списки командных скобок: `list` и `trivlist`, последний является частным случаем командных скобок `list`. Они достаточно гибкие, чтобы производить большинство списков, и используются в определении перечисленных выше командных скобок.

Самые используемые командные скобки, создающие списки, это:

- `itemize` — простое перечисление,
- `enumerate` — упорядоченное перечисление,
- `description` — перечисление с метками, задаваемыми пользователем.

Каждый пункт списка начинается с команды `\item`, необязательный аргумент которой дает метки пунктов.

4.3 Боксы

Бокс — это нечто, что TeX трактует как неделимую единицу, и поэтому не может разбивать между строками или страницами. Команда `\framebox` создает бокс в рамке, рисуя вокруг него горизонтальные и вертикальные прямые. Команда `\makebox` аналогична команде `\framebox` за тем исключением, что рамка вокруг результата обработки текста *text* не рисуется. Команда `\savebox` аналогична команде `\makebox` за тем исключением, что она не выдает никакого результата. Можно сказать, что полученный этой командой бокс помещается в *ячейку памяти* и может быть выдан позднее командой `usebox`. Имеются варианты команд `\newsavebox`, `\savebox` и `\usebox` с меньшим количеством аргументов.

В то время как команды `\framebox`, `\makebox` и `\savebox` обрабатывают свой аргумент *text* в LR моде, командные скобки `minipage` и команда `\parbox` обрабатывают этот аргумент в абзацной моде . Поэтому боксы, которые они производят, называются *абзацными боксами*.

[7] Пробелы такой величины вставляются перед командными скобками `center`, `description`, `enumerate`, `flushleft`, `flushright`, `itemize`, `quotation`, `quote`, `thebibliography`, `verse` и всеми командными скобками, создаваемыми с помощью декларации `\newtheorem`.

4.4 Таблицы

Таблицы легко получаются командными скобками `tabular`, описание которых имеется в каталоге. Для получения табулированной информации имеются также командные скобки `tabbing`.

5 Акценты

Полный список возможных акцентов приведен в следующей таблице:

Результат	Команда	Название
á	`\'a`	акут
b̆	`\u b`	краткость
ĉ	`\^c`	циркумфлекс
d̈	`\"d`	умлаут
ė	`\.e`	точка “над”
f̀	`\‘f`	гравис
ǧ	`\v g`	гачек
h̋	`\H h`	венгерский умлаут
ī	`\=\i`	макрон
j̃	`\~\j`	тильда
k̲	`\b k`	черта “под”
l̦	`\c l`	цедилла
m̤	`\d m`	точка “под”
n͡o	`\t no`	лига

Заметим, что все эти команды — кроме `\t` — делают акцент над или под только одним следующим за ними символом, следовательно, этот символ можно не заключать в фигурные скобки (хотя вреда от этого не будет). Обратите внимание, что команды `\i` и `\j` дают, соответственно, бесточечные i и j, которые и надо использовать с акцентами.

6 Плавающие вставки

Плавающая вставка в LaTeXe — это бокс, относительное положение которого во входном файле не определяет полностью его относительное положение в выходном результате. Плавающие вставки делаются двумя видами командных скобок — командными скобками `figure` и `table`. (У этих командных скобок имеются также и `*`-версии, а именно, `figure*` и `table*`, но они ведут себя точно так же, за исключением случая, когда в команде `\documentstyle` выбрана опция `twocolumn`.)

Команда `\caption` используется для того, чтобы получить подпись под таблицей или рисунком. Если Вы хотите сослаться на рисунок или таблицу, имеющую подпись, Вам надо вставить команду `\label`.

Имеется несколько параметров, которые воздействуют на расположение плавающих вставок в выходном документе. Сначала рассмотрим счетчики, которые влияют на это положение.

`bottomnumber` — максимальное количество плавающих вставок, которые могут помещаться внизу каждой страницы, содержащей и текст, и вставки.

`totalnumber` — максимальное количество плавающих вставок, которые могут помещаться на странице.

`topnumber` — максимальное количество вставок, которые могут помещаться вверху каждой страницы, содержащей и текст, и вставки.

Далее идут несколько параметров, значениями которых являются дробные числа между 0 и 1 и которые влияют на расположение вставок.

`\topfraction` — часть страницы, которая может быть занята вставками, расположенными вверху этой страницы (если она содержит и текст).

`\bottomfraction` — часть страницы, которая может быть занята вставками, расположенными внизу этой страницы (если она содержит и текст).

`\textfraction` — минимальная часть страницы (содержащей как текст, так и вставки), которая может быть занята текстом.

`\floatpagefraction` — минимальная часть страницы (содержащей только вставки), которая может быть занята вставками.

И наконец, параметры длины, действующие на внешний вид плавающих вставок:

`\floatsep` — величина вертикального пробела между вставками, расположенными на одной текстовой странице.

`\intextsep` — величина вертикального пробела, помещаемого над и под плавающей вставкой, которая находится в середине текстовой страницы при выборе опции `h`.

`\floatsep` — величина вертикального пробела между вставкой и текстом над или под ней, когда вставка помещается внизу или вверху страницы, содержащей текст и вставки.

7 Диаграммы и картинки

Простые картинки, состоящие из прямых, стрелок, окружностей и овалов, используются командные скобки `picture`. Для получения линейных диаграмм с помощью LaTeXa требуется задать длину линий, а также указать их положение на декартовой плоскости. Для этой цели используются не абсолютные единицы измерения — такие как дюймы или сантиметры — а относительные, значение которых задается в командных скобках `picture` параметром жесткой длины `\unitlength`.

Внутри командных скобок `picture` имеется только очень ограниченный выбор команд. Это команды `\put` и `\multiput` и такие декларации, как `\thicklines` и `\thinlines`. Все линии и формы внутри командных скобок `picture` рисуются командами `\line`, `\vector`, `\circle` и `\oval` и размещаются на декартовой плоскости командами `\put` и `\multiput`.

Заметим, что команды `\makebox`, `\framebox` и `\savebox` также можно использовать внутри командных скобок `picture`, но они ведут себя не так, как вне их.

8 Моды

При обработке входного файла LaTeX находится в одной их трех мод, а именно в абзацной моде, математической моде или в LR моде (LR — left to right). Даже когда вы используете командные скобки `picture`, а значит моду картинки — это просто частный случай LR моды.

8.1 Абзацная мода

Когда LaTeX обрабатывает обычный текст, он находится в абзацной моде. В TeXe имеется очень сложный алгоритм форматирования абзацев и принятия решения, где делать разбиения строк и вставлять переносы. Одно из следствий этого, то, что существует ограничение на величину абзацев, которые может содержать документ. Максимальный предел приблизительно

равен 1900 слов. Но есть возможность обмануть TEX и справиться с этой проблемой. Если у Вас есть абзацы более 1900 слов, вставьте кое-где в тексте примерно через 50 строк команды {\parfillskip=0pt\par\parskip=0pt\noindent}.

Чтобы указать LATEXу на то, что Вам надо начать новый абзац, можно просто вставить одну пустую строку или, что то же самое, два раза нажать клавишу CR (return); для той же цели служит командное слово или команда \par, которая сигнализирует о том, что следующее за ней начинается с нового абзаца. По умолчанию абзацы в выходном документе TEXа начинаются с небольшого отступа от левого поля. Величина отступа хранится в *параметре длины* \parindent и может быть изменена — скажем, на полдюйма — присваиванием \parindent=0.5in. Хотя это присваивание можно поместить в любом месте входного файла, лучше всего это сделать в преамбуле (поэтому на языке LATEXа это присваивание является *декларацией*; заметим, что знак равенства необязателен). Для того, чтобы TEX не делал отступ в первой строке абзаца, надо использовать команду\noindent. Величина добавочного вертикального пробела, который TEX вставляет между абзацами — в дополнение к обычному междустрочному пробелу — "хранится" в параметре длины \parskip.

8.2 Математическая мода

Огромное преимущество как TEXа, так и LATEXа над другими программами проявляется при наборе математических текстов. Чтобы получить математическую формулу или математическое выражение внутри текущего абзаца, ее надо заключить в знаки доллара. В LATEXе для этого имеются еще два альтернативных способа, но трудно даже придумать разумную причину, по которой их можно использовать, поскольку оба они сложнее прямолинейных знаков доллара.

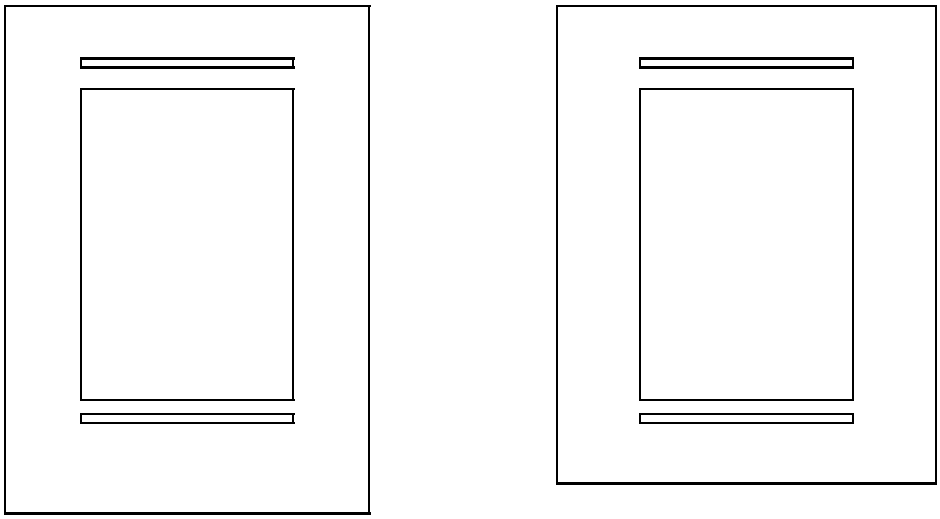
Между двумя одиночными знаками доллара TEX находится в *математической моде* (и обрабатывает там любую формулу в текстовом стиле). Пробелы внутри математической моды делаются совсем не так, как в абзацной, и вставка пробелов в математической моде не действует. Они имеют смысл в единственном месте — чтобы указать на конец командного слова, если после этого слова следует буква.

Чтобы выключить математическую формулу из текста, ее надо заключить в двойные знаки доллара. Чтобы получить это, в LATEXе имеется два альтернативных способа, а именно, \[x - y > 3\] и \begin{displaymath}x - y >3\end{displaymath}. Небольшое отличие между получением выключенной формулы с помощью двойных знаков доллара и двумя другими методами состоит в том, что если в команде \documentstyle Вы задаете опцию fleqn, выключенные формулы, полученные командными скобками displaymath или парой команд \[и \], не будут центрированы на странице — они будут отстоять от левого поля на расстояние, содержащееся в параметре длины \mathindent — в то время как выключенные формулы, полученные с использованием двойных знаков доллара, по-прежнему будут центрироваться. Так что если Вы намереваетесь использовать в команде \documentstyle опцию fleqn, то для выключенных формул пользуйтесь командами \[и \].

Внутри командных скобок displaymath, между командами \[и \] и между двойными знаками доллара TEX находится в *математической моде* (и обрабатывает формулы, которые там встречаются, в выключенном стиле). В математической моде TEX полностью игнорирует все пробелы.

При написании математических выражений не забывайте о правильной пунктуации. Во многих книгах одна из самых раздражающих вещей — это особенно относится к книгам некоторых специалистов по информатике — неправильная пунктуация в математических формулах.

В LATEXе имеется большое количество математических символов, перечисленных ниже.



Европейский формат A4 Американский формат $11 \times 8\frac{1}{2}$ дюймов

Рис. 1: Расположение заголовка, тела и основания страницы.

8.3 LR мода

LR мода — это в основном способ сохранить отрывок текста на строке одним куском. Некоторые, например, не хотят, чтобы в именах компьютерных программ делался перенос. Чтобы это не происходило, можно "запереть" их, например так: `\mbox{Имя_программы}`. Также хорошо позаботиться о том, чтобы математическая формула или выражение не были разбиты на строки. Например, написав `\mbox{$x - y > 3$}` можно с уверенностью сказать, что формула $x - y > 3$ поместится на одной строке. Внутри аргумента команды `\mbox` LaTeX находится в LR моде.

Команда `\mbox` (make box — делай бокс) производит *бокс*, что в TeXнологии означает блок материала, который трактуется как одно целое и ни при каких обстоятельствах не может быть разбит на составные части.

9 Стили страниц

Страница делится на три части: *заголовок*, *тело* и *основание*. Их размеры определяются параметрами стиля документа, а стиль страницы задает содержание заголовка и основания. Левые и правые страницы имеют различные значения параметров. В двустороннем стиле четные страницы будут левыми, а нечетные — правыми; в одностороннем стиле все страницы правые.

`\pagestyle{`*style*`}`

Декларация с обычными правилами для области действия, которая определяет стиль текущей страницы. Стандартными *style*-опциями являются следующие:

`plain` Заголовок пуст, а в основании есть только номер страницы. Это стиль страницы, устанавливаемый по умолчанию.

`empty` Заголовок и основание страницы пустые.

`headings` Заголовок содержит информацию, определяемую стилем документа (обычно это заголовок единицы рубрикации) и номер страницы; основание пустое.

`myheadings` То же, что `headings`, за тем исключением, что информация в заголовке определяется командами `\markboth` и `\markright`, описанными ниже.

`\thispagestyle`

То же, что `\pagestyle`, с тем исключением, что эта команда применяется только к текущей странице. Это глобальная декларация.

`\markright{`*right_head*`}`
`\markboth{`*left_head*`}{`*right_head*`}`

Эти команды задают информацию для стилей страницы `headings` и `myheadings`:

`\pagenumbering{`*num_style*`}`

Задает стиль номеров страниц. Это глобальная декларация. Возможными значениями *num_style* являются:

`arabic`	арабские цифры.
`roman`	римские цифры нижнего регистра.
`Roman`	римские цифры верхнего регистра.
`alph`	буквы нижнего регистра.
`Alph`	буквы верхнего регистра.

Команда `\pagenumbering` переопределяет `\thepage` в `\`*num_style*`{page}`.

`\twocolumn{`*текст*`}`

Начинает новую страницу и начинает печать в два столбца. Если присутствует аргумент *текст*, то он набирается в абзацном боксе шириной в два столбца наверху новой страницы.

`\onecolumn`

Начинает новую страницу и печать в один столбец.

9.1 Параметры стиля

Следующие параметры обычно изменяются только в преамбуле. Если их изменить в середине документа, могут возникнуть аномалии.

`\oddsidemargin`	`\evensidemargin`	`\marginparwidth`
`\marginparsep`	`\topmargin`	`\headheight`
`\headsep`	`\textheight`	`\textwidth`
`\topskip`	`\footheight`	`\footskip`

9.2 Изменение размера выходной страницы

Когда LaTeX готовит страницу выходного результата, он может помещать текст в три ее различные области, а именно, в заголовок, тело и основание. Если создается бегущий заголовок, он вместе с текущим номером страницы отправляется в область ее заголовка; если же бегущий заголовок отсутствует, текущий номер страницы помещается в центре ее области основания. Заметим, что сноски, если они имеются, располагаются в области *тела* выходной страницы. На рис.1, стр.17[8] показано размещение этих трех областей на листе бумаге размера $11 \times 8\frac{1}{2}$ дюйма и на формате A4 в предположении, что задан стиль документа `article` и опция `10pt` или `11pt`. На этих схемах внешний прямоугольник — в обоих случаях это физический лист бумаги, а три внутренних прямоугольника (сверху вниз) — это заголовок, тело и основание. Заметим, что на листе формата A4 тело области текста не центрировано. Это происходит потому, что TeX и LaTeX предполагают, что Вы будете печатать свой выходной результат скорее на стандарте, принятом в Америке, чем на формате A4, стандартном для Европы.

Существуют команды для различного размещения тела текста на физической странице, а также для изменения их размеров. Например, расстояние от верхней границы физического листа бумаги до верхней границы области заголовка равно одному дюйму плюс значение параметра длины LaTeXa `\topmargin`. Это можно изменить присваиванием вида `\topmargin=25in`. Заметим, что изменение значения `\topmargin` приводит к сдвигу вниз или вверх и заголовка страницы, и ее тела, и основания, поскольку они рассматриваются как одно целое. Расстояние от левой границы физического листа бумаги до левой границы ее области тела равно одному дюйму плюс значение параметра жесткой длины `\oddsidemargin`.[9] Можно также изменить величину тела страницы, меняя значения параметров длины LaTeXa `\textheight` и `\textwidth`. Это делается такими присваиваниями:

```
\textheight=8.5in
\textwidth=6in
```

Эти присваивания не меняют размеры левого или верхнего полей.

9.3 Подготовка текста в две колонки

Чтобы получить документ, на страницах которого текст расположен в две колонки, надо в `\documentstyle` задать опцию `twocolumn`. Большинство команд при печати в две колонки работают так же, как и при печати в одну колонку. Однако при выборе опции `twocolumn` параметрам длины присваиваются начальные значения, которые отличаются от значений при печати в одну колонку. Например, изменяется величина абзацного отступа и отступа в списках.

Если в две колонки должна набираться часть документа, используется декларация `\twocolumn`; декларация `\onecolumn` возвращает к набору в одну колонку. Как `\twocolumn`, так и `\onecolumn` приводят к тому, что печать начинается с новой страницы. Важное отличие опции `twocolumn` от декларации `\twocolumn` в том, что последняя не меняет значений параметров длины, как это делает первая.

При задании опции `twocolumn` меняет свое поведение лишь малое количество команд, среди них команды `\newpage`, `\pagebreak` и `\marginpar`. Команда `\marginpar` помещает заметку

[8] Размер листа бумаги формата A4 равен 297×210 мм. Стиль документа `article` является *односторонним* стилем. В одностороннем стиле документа все страницы (концептуально) являются правосторонними или нечетнонумеруемыми. Это означает, что заголовок, тело и основание как нечетных, так и четных страниц на физическом листе бумаги располагаются относительно левой и верхней границы страницы одинаково.

[9] При двусторонней печати на страницах с нечетными номерами на левые поля действует параметр длины `\oddsidemargen`, а на величину левого поля страницы с четными номерами — параметр длины `\evensidemargin`. При односторонней печати и в том и в другом случае действует только параметр `\oddsidemargin`.

на полях на ближайшем поле, т.е. на правом поле, если команда `\marginpar` встречается в правой колонке, и на левом, если она встречается в левой колонке, причем выбор односторонней или двухсторонней печати не влияет на расположение таких заметок. Команды `\pagebreak` и `\newpage` начинают не новую страницу, а новую колонку, а команды `\clearpage` и `\cleardoublepage`, по-прежнему, обрывают текущую страницу.

Параметр длины `\columnsep` равен величине промежутка, разделяющего две колонки текста, а `\columnseprule` — толщине вертикальной линии, помещаемой между колонками (по умолчанию она равна нулю дюймам).

10 Определение собственных команд

Одна из причин для определения своих собственных команд, известных в TEXнике как макрокоманды или макросы — это экономия времени для набора. Эти определения используют примитивную команду TEXа `\def`; в LATEXе же для определения макрокоманд служит команда `\newcommand`. Одно из основных отличий между командами `\def` и `\newcommand` состоит в том, что команда `\newcommand` проверяет, не существует ли уже команда, которую Вы пытаетесь определить; если да, то Ваша попытка определения отвергается. Наоборот, команда `\def` с успехом переопределяет существующую команду.[10] Некоторые относят это к преимуществам декларации LATEXа `\newcommand`, поскольку ею нельзя случайно переопределить важную команду TEXа или LATEXа, а случайное переопределение некоторых примитивных команд TEXа может привести к драматическим изменениям полученного выходного результата, поскольку эта команда могла быть использована в определении многих других команд.

Есть возможность иметь в файле локальные определения — их область действия ограничивается фигурными скобками. Фигурные скобки, следующие за списком аргументов, являются частью синтаксиса команды `\def` и не ограничивают области действия деклараций, которые могут встретиться в определении. Так, если Вы определяете `\def\B{\bf B}`, где `\bf` — это декларация, которая задает в выходном документе жирный шрифт, то при использовании команды `\B` жирным шрифтом напечатается все, что следует за этой командой. Наоборот, при использовании декларации `\def\B{{\bf B}}` можно быть уверенным, что команда `\B` напечатает жирным шрифтом только букву `B`. Это относится и к механизму макроопределений LATEXа.

Определения, вводимые командами `\def`, `\newcommand` или `\renewcommand` могут располагаться в любом месте входного файла, но если Вы часто используете LATEX, их удобно поместить в свой собственный стилевой файл. Он может иметь любое основное или первое имя, но обязательно расширение `sty`. Для иллюстрации предположим, что Ваш стилевой файл называется `own.sty`. В него Вы должны поместить, как минимум, все Ваши нелокальные определения. Затем для того, чтобы эти определения загружались, когда Вы используете LATEX, Вам надо включить первое или основное имя этого стилевого файла в список опций *opt-list* команды`\documentstyle` следующим образом:

```
\documentstyle[11pt,own]{article}
```

В то время как имена всех команд, которые Вы определили в своем входном файле, должны состоять только из букв, имена команд, определенных в стилевом файле, могут содержать знак `@`; так, `\@lqq` и `\@rqq` допустимы в качестве имен команд, определенных в стилевом файле. Однако такие команды вне стилевых файлов использовать нельзя, хотя команды, определенные в одном стилевом файле, можно использовать в другом.

Как уже упоминалось, локальными называются те определения, область действия которых ограничивается фигурными скобками. Наоборот, *глобальными* называются такие декларации, область действия которых так не ограничивается. Если Вы хотите поместить глобальное определение внутри фигурных скобок, надо перед командой `\def` поставить команду `\global`.

[10] В LATEXе для переопределения уже существующей команды служит `\renewcommand`.

Один из недостатков команд LaTeXa `\newcommand` и `\renewcommand` — это то, что они всегда делают *неглобальные* определения.

11 Математические символы

В TeXe имеются несколько категорий математических символов, наиболее важные из которых показаны в следующей таблице:

Вид атома	Имя	Пример	Команда
ординарный	Ord	`\gamma`	`\mathord`
большой оператор	Op	`\prod`	`\mathop`
бинарный оператор	Bin	`\wedge`	`\mathbin`
отношение	Rel	`\gg`	`\mathrel`
открывающий	Open	`\{`	`\mathopen`
закрывающий	Close	`\}`	`\mathclose`
пунктуация	Punct	`,`	`\mathpunct`
внутренний	Inner	`{1 \over 2}`	`\mathinner`

11.1 Ординарные символы

TeX в математической моде считает ординарными символами 26 прописных латинских букв от A до Z и 26 строчных латинских букв от a до z, а также 18 следующих символов:

```
0 1 2 3 4 5 6 7 8 9 ! ? | / ‘ @ "
```

(*В русифицированном* LaTeXe *ординарными символами также считаются прописные и строчные русские буквы.*) Выражение "символ является ординарным" означает, что TeX, когда печатает такие символы один за другим, не помещает между ними никаких дополнительных пробелов. Заметим, что точка в математической моде считается не знаком пунктуации, а ординарным символом. Это приводит к тому, что действительные числа в математической моде набираются правильно. Так, выражение $12.72 > x$ было получено командами `$12.72 > x$`.

Когда любая из этих прописных или строчных букв встречается внутри математических скобок (внутри `$` либо `$$`), то она набирается математическим курсивом. Этот шрифт несколько отличается от текстового курсива. Буквы его несколько крупнее и совершенно другие пробелы. Это становится понятно, если посмотреть на слово "effluent", которое в текстовом курсиве выглядит как *effluent*, а в математическом курсиве — как $effluent$. Причина этого в том, что математики обычно обозначают одной буквой имена функций и переменных, а буквы, расположенные рядом, обозначают умножение. С другой стороны, программистам нравятся многобуквенные идентификаторы. Такой идентификатор в TeXe можно получить командой `${\it effluent}$`.

11.1.1 Строчные греческие буквы

Греческий алфавит состоит из 24 букв, строчные формы которых получаются так:

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

TEX рассматривает их как ординарные символы. Заметим, что омикрон в математической моде получается вводом “o”. Кроме того, математики используют варианты букв дельта, эпсилон, тета, пи, ро и фи:

∂	`\partial`	ϑ	`\vartheta`	ϱ	`\varrho`
ε	`\varepsilon`	ϖ	`\varpi`	φ	`\varphi`

Имеется еще один вариант эпсилон, а именно, $\in$, получаемый командой `\in` и используемый для обозначения члена множества, но TEX трактует его как отношение, а не как ординарный символ.

TEX также имеет символ ς, получаемый командой `$\varsigma$`. Этот символ, а также упсилон, не являются буквами, которые используются математиками. Они включены в TEX на случай, если кому-нибудь захочется привести короткую греческую цитату. Если же Вам нужна греческая версия TEXа, то придется задать новый шрифт.

11.1.2 Прописные греческие буквы

A	`{\rm A}`	H	`{\rm H}`	N	`{\rm N}`	T	`{\rm T}`
B	`{\rm b}`	Θ	`\Theta`	Ξ	`\Xi`	Υ	`\Upsilon`
Γ	`\Gamma`	I	`{\rm I}`	O	`{\rm O}`	Φ	`\Phi`
Δ	`\Delta`	K	`{\rm K}`	Π	`\Pi`	X	`{\rm X}`
E	`{\rm E}`	Λ	`\Lambda`	P	`{\rm P}`	Ψ	`\Psi`
Z	`{\rm Z}`	M	`{\rm M}`	Σ	`\Sigma`	Ω	`\Omega`

Заметим, что по умолчанию получаются “прямостоящие” версии прописных греческих букв, поскольку именно они наиболее часто используются в математике.

11.1.3 Прописные греческие курсивные буквы

A	`A`	I	`I`	P	`P`
B	`B`	K	`K`	Σ	`{\mit\Sigma}`
Γ	`{\mit\Gamma}`	Λ	`{\mit\Lambda}`	T	`T`
Δ	`{\mit\Delta}`	M	`M`	Υ	`{\mit\Upsilon}`
E	`E`	N	`N`	Φ	`{\mit\Phi}`
Z	`Z`	O	`O`	Ψ	`{\mit\Psi}`
Θ	`{\mit\Theta}`	Π	`{\mit\Pi}`	Ω	`{\mit\Omega}`

11.1.4 Прописные каллиграфические буквы

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

11.1.5 Дополнительные символы

Таблица различных символов типа Ord:

$\forall$	`\forall`	ℓ	`\ell`	$\flat$	`\flat`
$\exists$	`\exists`	$\wp$	`\wp`	$\sharp$	`\sharp`
$\neg$	`\neg`	$\Re$	`\Re`	$\natural$	`\natural`
$\top$	`\top`	$\Im$	`\Im`	$\spadesuit$	`\spadesuit`
$\bot$	`\bot`	$\prime$	`\prime`	$\heartsuit$	`\heartsuit`
$\emptyset$	`\emptyset`	∇	`\nabla`	$\diamondsuit$	`\diamondsuit`
$\aleph$	`\aleph`	$\surd$	`\surd`	$\clubsuit$	`\clubsuit`
∞	`\infty`	$\|$	`|` или `\Vert`	$\triangle$	`\triangle`
$\hbar$	`\hbar`	$\angle$	`\angle`	$\diamond$	`\Diamond`
$\imath$	`\imath`	$\backslash$	`\backslash`	$\Box$	`\Box`
$\jmath$	`\jmath`	$\mho$	`\mho`	$\vert$	`\vert` или `\vert`
/	`/`	%	`\%`	$	`\$`
#	`\#`	_	`\_`	&	`\&`

Команды для символов $\diamond$, $\Box$ и $\mho$ — это специфичные команды LaTeXa, в plain TeXe их нет.

11.2 Символы больших операторов

Таблица различных символов больших операторов:

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

Когда в выключенном стиле к управляющим словам для больших операторов добавляются нижние и верхние индексы, они появляются, соответственно, под и над этими символами в виде пределов. Например, `$$\sum_{i=1}^{i=n} i^2,$$` дает

$$\sum_{i=1}^{i=n} i^2,$$

в то время как внутри текстовой строки `$\sum_{i=1}^{i=n} i^2$` дает $\sum_{i=1}^{i=n} i^2$. Заметим, что величина оператора тоже изменилась.

11.2.1 Символы элементарных функций

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

Эти операторы распадаются на две категории, различающиеся в зависимости от того, как в выключенном стиле ведут себя их нижние и верхние индексы. Нижние и верхние индексы в выключенном стиле набираются в виде пределов, когда они присоединены к следующим операторам: `\det`, `\gcd`, `\inf`, `\lim`, `\liminf`, `\limsup`, `\max`, `\min`, `\Pr` и `\sup`.

11.2.2 Определение собственных больших операторов

Для того, чтобы определить свои собственные функции типа log, надо решить, как с ними должны себя вести нижние и верхние индексы. Если Вы хотите, чтобы в выключенном стиле они вели себя как пределы, то надо определить свой оператор так:

```
\def\dom{\mathop{\rm dom}\limits}
```

В противном случае определите его так:

```
\def\dom{\mathop{\rm dom}\nolimits}
```

Например, чтобы получить формулу

$$\IF_{i=1}^{n} b_i P_i = \IF_{i=1}^{n} b_i^* P_i.$$

средствами LaTeXa, надо ввести

```
$$\IF_{i=1}^n b_i P_i = \IF_{i=1}^n b_i^* P_i.$$
```

где `\IF` определено как `\def\IF{\mathop{\rm IF}\limits}`.

11.3 Символы бинарных операторов

К бинарным операторам относятся символы типа $+$ или $\sqcap$. Они составляют член или выражение из двух членов или выражений. Следует отличать бинарный оператор от символа бинарного отношения типа $>$ или $\sqsubset$, поскольку бинарные отношения составляют из двух членов или выражений некоторое утверждение или предположение. Это различие отражается в величине пробелов, отделяющих операторы от их аргументов. В случае с отношениями, например, как в

$$x > y,$$

аргументы отношения отделяются толстыми пробелами. Их величина представлена командой `\;` и равна около пяти восемнадцатых квадрата.[11] У операторов, например, как в

$$x + y,$$

аргументы от оператора отделяются средними пробелами. Их величина представлена командой `\:` и равна около двух девятых квадрата.[12]

[11] Квадрат равен одному em.

[12] В TeXe такой пробел получается командой `\>`, которая в командных скобках LaTeXa `tabbing` имеет специальное значение.

Приведем таблицу различных символов бинарных операторов:

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

Некоторых из этих символов, а именно символов $\lhd$, $\rhd$, $\unlhd$ и $\unrhd$, в TEXe нет. Эта таблица полная.

11.3.1 Определение собственных символов операторов

Скажем, Вы хотите определить символ $+\!\!+$, который некоторые авторы используют для конкатинации списков. Это можно сделать следующим образом:[13]

```
\def\con{\mathbin{+\mkern-8mu+}}
```

Такое определение способствует правильному распределению пробелов:

$$x +\!\!+ (y +\!\!+ z) = (x +\!\!+ y) +\!\!+ z.$$

Эта формула была получена так:

```
$$x \con (y \con z) = (x \con  y) \con z.$$
```

11.4 Символы бинарных отношений

Таблица различных символов бинарных отношений:

$>$	`>`	$<$	`<`	$=$	`=`
$\leq$	`\leq`	$\geq$	`\geq`	$\equiv$	`\equiv`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succceq`	$\simeq$	`\simeq`
$\ll$	`\ll`	$\gg$	`\gg`	$\asymp$	`\asymp`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\smile$	`\smile`	$\frown$	`\frown`	$\doteq$	`\doteq`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$:$	`:`				

[13] Здесь mu (math unit) — это единица длины. В одном em содержится 18 mu. Единицу mu можно использовать только в математической моде.

Символов $\sqsubset$, $\sqsupset$ и $\bowtie$ в plain TEXe нет. Отметим также, что : TEX трактует как отношение. Это обеспечивает правильные пробелы в такой, например, конструкции: `$$x := x + 1.$$`, которая дает

$$x := x + 1.$$

Чтобы получить двоеточие, которое используется для указания типа объекта или функции, посмотрите описание символов пунктуации ниже. Чтобы получить отрицающий вариант некоторого символа отношения, надо просто поставить перед ним `\not`. Так, `$x \not\equiv y$` дает $x \not\equiv y$.

11.4.1 Стрелки

Все следующие далее символы стрелок TEX считает бинарными отношениями и распределяет пробелы соответственно этому. Таблица символов стрелок:

$\uparrow$	`\uparrow`	$\downarrow$	`\downarrow`
$\Uparrow$	`\Uparrow`	$\Downarrow$	`\Downarrow`
$\updownarrow$	`\updownarrow`	$\Updownarrow$	`\Updownarrow`
$\nearrow$	`\nearrow`	$\searrow$	`\searrow`
$\swarrow$	`\swarrow`	$\nwarrow$	`\nwarrow`
$\leftarrow$	`\leftarrow`	$\longleftarrow$	`\longleftarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`
$\rightarrow$	`\rightarrow`	$\longrightarrow$	`\longrightarrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`
$\rightleftharpoons$	`\rightleftharpoons`	$\leadsto$	`\leadsto`

Символа $\leadsto$ в plain TEXe нет. Заметим, что после команды `\left` или `\right` команды `\downarrow`, `\Downarrow`, `\uparrow`,`\Uparrow`, `\updownarrow` и `\Updownarrow` дают ограничители, размер которых зависит от того, что они ограничивают.

11.5 Ограничители

11.5.1 Открывающие символы

$($	`(`	$[$	`[`	$\{$	`{`
$\langle$	`\langle`	$\lfloor$	`\lfloor`	$\lceil$	`\lceil`

Символ { можно также получить командой `\lbrace`, а символ [— командой `\lbrack`.

11.5.2 Закрывающие символы

$)$	`)`	$]$	`]`	$\}$	`}`
$\rangle$	`\rangle`	$\rfloor$	`\rfloor`	$\rceil$	`\rceil`

Символ } можно также получить командой `\rbrace`, а символ] — командой `\rbrack`.

11.5.3 Определение собственных ограничителей

Некоторые авторы используют < и > в качестве скобок, но в этом случае пробелы будут расставлены неправильно. Чтобы использовать такие символы в качестве ограничителей, надо определить их как `\def\lacute{\mathopen{<}}` и `\def\racute{\mathopen{>}}`. Обратите внимание на следующий пример, в котором операторы слева от знака iff были переопределены.

$$<x, y> \iff < x, y > .$$

Это было получено так:

```
$$\lacute x, y \racute \iff < x, y>.$$
```

Сравните это со следующей формулой:

$$<x, y> \iff \langle x, y\rangle,$$

которая была получена так:

```
$$\lacute x, y \racute \iff \langle x, y \rangle,$$
```

11.6 Символы пунктуации

Когда в математических формулах встречаются запятые или точки с запятыми, TEX помещает после них тонкий пробел. Так же он поступает и с двоеточием, если оно получено командой `\colon`. Обратите внимание на различные пробелы в следующих выражениях:

$$f\colon x \to y \qquad f : x \to y,$$

которые были получены так:

```
$$f \colon x \rightarrow y \qquad f : x \rightarrow y,$$
```

Следующие символы можно использовать в любой моде, но в математической моде они считаются символами пунктуации: † (`\dag`),‡ (`\ddag`), § (`\S`), ¶ (`\P`), © (`\copyright`) и £ (`\pounds`).

11.7 Составные конструкции

11.7.1 Акценты

В математических работах принято дополнять символы — особенно, переменные — различного вида акцентами. В математической моде TEX предусматривает следующие виды акцентов:

Результат	Ввод	Результат	Ввод
$\acute{a}$	`\acute a`	$\dot{f}$	`\dot f`
$\bar{b}$	`\bar b`	$\grave{g}$	`\grave g`
$\breve{c}$	`\breve c`	$\hat{h}$	`\hat h`
$\check{d}$	`\check d`	$\tilde{\imath}$	`\tilde \imath`
$\ddot{e}$	`\ddot e`	$\vec{\jmath}$	`\vec \jmath`

Эта таблица также иллюстрирует применение команд `\imath` и `\jmath` для получения в математической моде бесточечных i и j. Под акцентами надо использовать именно их.

Все указанные выше акценты имеют один размер и если поместить любой из них над выражением, состоящим из более чем одной буквы, TEX просто отцентрирует этот акцент над выражением в целом. Однако есть две команды для получения акцентов, величина которых меняется в зависимости от размера выражения, к которому они относятся. Это команды `\widehat` и `\widetilde`; каждая из них может давать акценты трех размеров.

11.7.2 Подчеркивание, надчеркивание и связки

Этими способами в TEXe можно украшать выражения любого размера, что иллюстрируется в следующей таблице:

Результат	Ввод
$\overline{\phi \land \psi}$	`\overline{\phi \land \psi}`
$\underline{\phi \land \psi}$	`\underline{\phi \land \psi}`
$\overrightarrow{\phi \land \psi}$	`\overrightarrow{\phi \land \psi}`
$\overleftarrow{\phi \land \psi}$	`\overleftarrowe{\phi \land \psi}`
$\overbrace{\phi \land \psi}$	`\overbrace{\phi \land \psi}`
$\underbrace{\phi \land \psi}$	`\underbrace{\phi \land \psi}`

11.7.3 Нижние и верхние индексы

Чтобы получить выражение с нижним индексом типа x_i, надо во входном файле поставить `$x_i$`. Для получения нижнего индекса вместо символа подчеркивания можно также использовать команду `\sb`. Если нижний индекс представляет собой выражение, состоящее из нескольких символов, то это выражение надо заключить в фигурные скобки. Верхние индексы получаются либо символом `^`, либо командой `\sp`. С соответствующими изменениями, все, что говорилось выше для нижних индексов, справедливо и для верхних.

Нижние и верхние индексы, которые вы добавляете к символам, сами могут иметь нижние и верхние индексы. Заметим, что для индексов используются символы, размеры которых меньше обычно применяющихся в формулах (и в тексте, и в выключенных). Это относится не только к буквам и цифрам, но и к большинству других символов. Если индексы сами имеют нижние или верхние индексы, то размер этих вторых индексов будет еще меньшим, но индексы следующего уровня меньшими уже не будут: TEX для набора математических формул оперирует только тремя размерами. Чтобы заставить TEX брать не тот размер шрифта, который принят по умолчанию, можно использовать команды `\textstyle`, `\scriptstyle` и `\scriptscriptstyle`.

11.7.4 Дроби и подобные им структуры

Дроби в LATEXe можно получить различными способами. Можно использовать символ слэша `/`. В то время как символы сложения ($+$), умножения ($\times$) и деления ($\div$) считаются бинарными операциями, TEX трактует `/` как ординарный символ, поскольку, когда он обозначает деление, в практике набора *не* принято помещать вокруг него дополнительные пробелы. Дроби, использующие горизонтальную черту, получаются либо командой LATEXa `\frac`, либо командой примитивного TEXa `\over`. В действительности команды plain TEXa `\choose`, `\brack` и `\brace` определяются через команду примитивного TEXa `\atopwithdelims`.

11.8 Многострочные конструкции

11.8.1 Командные скобки `array`

В LATEXe командные скобки `array` дают очень гибкий способ получения такого регулярного двумерного расположения математических символов, как таблицы и матрицы. (Слово “ре-

гулярное" предполагает, что символы или матрицы расположены довольно естественным способом в виде рядов и/или колонок. Если это не так, то лучше использовать командные скобки `picture`).

Заметим, что командные скобки `array` очень похожи на командные скобки `tabular`, и все возможности командных скобок `tabular` также имеются и у командных скобок `array`. Основное отличие между ними состоит в том, что командные скобки `array` можно использовать только в математической моде и элементы получаемых структур тоже (как правило) обрабатываются в математической моде.

11.8.2 Матрицы в TEXe

По нашему мнению, команда plain TEXa `\matrix` и родственные ей команды намного удобнее для получения матриц, чем командные скобки `array` в LATEXe.[14]

Поскольку часто встречаются матрицы, окруженные большими круглыми скобками, plain TEX для получения их имеет команду `\pmatrix`. В plain TEXe также есть команда `\bordermatrix` для получения матриц с метками.

11.8.3 Массивы уравнений и помеченные формулы

Командные скобки `eqnarray*` производят массив уравнений, ни одна строка которого не помечена. Внутри командных скобок `eqnarray*` (или `eqnarray`) можно использовать команду `\lefteqn`, чтобы получить формулу, ширину которой LATEX считает равной нулю.

Чтобы получить одну выключенную математическую формулу с числовой меткой, надо использовать командные скобки `\equation`. Заметим, что метка формулы генерируется автоматически. Она будет получена, даже если команда `\label` отсутствует. Функция команды `\label` — облегчить ссылку на эту формулу из других частей написанного Вами текста. Ссылаться на помеченную формулу, содержащую команду `\label{`*key*`}`, надо командой `\ref{`*key*`}`. Если вы хотите сослаться на страницу, на которой встречается эта формула, то надо использовать команду `\pageref`; например, `p.~\pageref{AA}` производит "p. 100". Здесь тильда вставляет обычный пробел между словами, но не позволяет делать разрыв строки. (*Прим. переводчиков: в русском языке в выражении типа "стр.100" пробел после точки не ставится, поэтому тильда не нужна.*)

По умолчанию метки формул, которые автоматически создает LATEX, размещаются на выходной странице у ее правого поля. Если вы хотите, чтобы вместо этого метки размещались у левого поля страницы, следует в команде `\documentstyle` использовать опцию `leqno`.

Чтобы получить несколько выключенных нумерованных уравнений, можно использовать командные скобки `eqnarray`. Команда `\nonumber` подавляет автоматическое получение метки на той строке, на которой она встречается.

12 Выбор шрифта

12.1 Изменение стиля шрифта

Следующие декларации выбирают указанный стиль шрифта:

`\rm`	Романский	`\it`	*Курсив*	`\sc`	Капитель
`\em`	*Выделительный*	`\sl`	*Наклонный*	`\tt`	`Машинописный`
`\bf`	**Жирный**	`\sf`	Рубленый		

[14] Команды plain TEXa, упомянутые в этой секции, можно вставлять во входной файл LATEXa, и они будут правильно обработаны.

Если стиль шрифта отсутствует в текущем размере, то декларация выбирает подходящий стиль и печатает предупреждение на терминале. Ниже указаны ограничения на использование этих команд в математической моде.

12.2 Изменение размера шрифта

Следующие декларации выбирают размер шрифта, а также выбирают прямой стиль этого размера. Они перечислены в порядке возрастания размера; в некоторых стилях документа две декларации могут иметь одинаковый результат.

`\tiny`	`\small`	`\large`	`\huge`
`\scriptsize`	`\normalsize`	`\Large`	`\Huge`
`\footnotesize`		`\LARGE`	

Эти команды не могут быть использованы в математической моде.

12.3 Загрузка шрифтов

`\newfont{`*cmd*`}{`*font_name*`}`

Определяет имя команды *cmd* (которая должна быть неопределенной в текущий момент) как декларацию, выбирающую в качестве текущего шрифта шрифт, называемый *font_name*. Вновь определяемая команда *cmd* не может быть использована в математической моде.

`\symbol{`*num*`}`

Выбирает символ с номером *num* из текущего шрифта. Восьмеричным (основание 8) и шестнадцатеричным (основание 16) числам предшествуют соответственно `'` и `"`.

12.4 Шрифты в математической моде

LaTeX располагает 10 различными размерами и 8 различными стилями шрифта, включая математический курсив. Каждой из 80 комбинаций размера/стиля соответствует отдельный шрифт. Эти шрифты делятся на три класса: *предварительно загружаемые*, *загружаемые по требованию* и *не имеющиеся в наличии*. Когда запрашивается не имеющийся в наличии шрифт, то вместо него берется другой, который может быть загружен предварительно или по требованию, и на терминале печатается предупреждение. Шрифты, загружаемые предварительно и по требованию, выглядят одинаково в абзацной моде и LR моде, и иначе — в математической моде.

Комбинация размер/стиль, которая соответствует шрифту, загружаемому по требованию, может действовать неправильно, когда она используется в математической моде; тогда печатаются либо символы неправильного размера, либо вообще никакие символы не печатаются, и генерируется одно из следующих сообщений об ошибке:

```
! \textfont         ... is undefined (character ...).
! \scriptfont       ... is undefined (character ...).
! \scriptscriptfont ... is undefined (character ...).
```

Правила, точно описывающие случай, когда возникает эта проблема, слишком сложные, но решение простое: используйте команду, имеющую вид

`\load{`*size*`}{`*style*`}`

где *size* — команда, изменяющая размер, а *style* — команда стиля шрифта, и обе они задают нужный шрифт. Команда `\load` должна предшествовать первому использованию шрифта в математической моде, и она не должна находиться внутри фигурных скобок или командных скобок.

В математической моде имеется четыре математических стиля: *выключенный* (display), *текстовый* (text), *индексный* (script) и *повторного индекса* (scriptscript). Выключенный и текстовый стили отличаются в основном размером некоторых символов и размещением нижних индексов у некоторых символов и функций. Индексный стиль используется для нижних и верхних индексов, а стиль повторных индексов — для нижних и верхних индексов последующих уровней.

Для каждой комбинации стиля/размера в математической моде требуются три шрифта: один для стилей выключенного и текстового, один для индексного стиля и один для стиля повторных индексов. В идеале эти шрифты должны быть различных размеров, за исключением случаев, когда в результате получился бы слишком мелкий шрифт, который невозможно читать. Однако выбор шрифтов ограничивается в LaTeXe двумя правилами:

(1) в стиле индекса и повторного индекса могут быть использованы только предварительно загруженные шрифты

(2) комбинация стиль/размер, соответствующая шрифту, загружаемому по требованию, использует один и тот же шрифт для всех математических стилей.

Это значит, что для некоторых комбинаций стиля/размера нижние и верхние индексы могут печататься в слишком большом шрифте. Требуемый математический стиль выбирают следующие декларации:

```
\displaystyle
\textstyle
\scriptstyle
\scriptscriptstyle
```

Декларация `\boldmath` выбирает жирный математический курсив и шрифты с жирными математическими символами. При этом буквы, цифры и большинство символов, используемых в математической моде, печатаются жирным шрифтом (см. ее описание в каталоге). Однако символы, которые составляются путем комбинации двух других символов, такие, например, как символ $\Longrightarrow$ (`\Longrightarrow`), составленный из $=$ и $\Rightarrow$, могут давать неправильные результаты.

Ниже перечислено то, что не печатается жирным шрифтом командой `\boldmath`.

- Текст, который печатается в индексном стиле или стиле повторного индекса (обычно верхние и нижние индексы).
- Текст, создаваемый следующими символами ввода:

  ```
  +  :  ;  !  ?  (  )  [  ]
  ```

- Символы переменного размера.
- Большие ограничители, создаваемые командами `\left` и `\right`. Однако ограничители нормального размера (не круглые и не квадратные скобки, создаваемые командами `\left` и `\right`), печатаются жирным шрифтом.

Декларация `\unboldmath` отменяет действие команды `\boldmath`. Ни одна из этих команд не может быть использована в математической моде.

```
\begin{thebibliography}{7}
\bibitem{hal:towards}
Michael Hallett,
``Towards a Theory of Mathematical Research Programmes (I)'',
{\it British Journal for the Philosophy of Science},
vol.~30 (1979), pp.~1--25.
%
 \bibitem{lak:proofs}
Imre Lakatos,
{\it Proofs and Refutations:
The Logic of Mathematical Discovery},
[Cambridge, Cambridge University Press, 1976].
\end{thebibliography}
```

References

[1] Michael Hallett, "Towards a Theory of Mathematical Research Programmes (I)", *British Journal for the Philosophy of Science*, vol. 30 (1979), pp. 1–25.

[2] Imre Lakatos, *Proofs and Refutations: The Logic of Mathematical Discovery*, [Cambridge, Cambridge University Press, 1976].

Рис. 2: Использование командных скобок `thebibliography`.

13 Библиографии и алфавитные указатели

В LaTeXe библиографию можно сделать несколькими различными способами, два из которых разъясним здесь: первый использует командные скобки `thebibliography`, а второй привлекает дополнительную систему, так называемый BibTeX.

13.1 Командные скобки `thebibliography`

Пример использования командных скобок `thebibliography` показан в верхней части рис.2, стр.32, а результат, полученный после обработки его LaTeXом — в нижней части этого рисунка (в русифицированном LaTeXe заголовком библиографии будет "Список литературы"). Открывающая команда командных скобок `thebibliography` требует обязательного аргумента, который используется, чтобы определить внешний вид полученного выходного результата. Например, если Вы ссылаетесь менее чем на 10 источников, то для этого аргумента можно использовать одну цифру[15]. Если Вы ссылаетесь на число источников от 10 до 99, то в этом аргументе должно быть две цифры, и т.д.

Элементы библиографии в командных скобках `thebibliography` вводятся командой `\bibitem{`*key*`}`, где аргумент *key* (ключ) состоит из букв, цифр и знаков препинания, кроме запятой. Для ссылок на элемент библиографии следует использовать команду `\cite{`*key*`}`. LaTeX автоматически генерирует числовые перекрестные метки. Так, исходя из входных данных, показанных в верхней части рис.2, стр.32, `\cite{lak:proofs}` генерирует [2], а `\cite{hal:towards}` генерирует [1]. Однако чтобы получить в выходном результате пере-

[15] Все цифры имеют одну и ту же ширину, а именно равную половине em текущего шрифта, поэтому не имеет значения, какую цифру использовать.

```
\begin{thebibliography}{76}
\bibitem[Hallett 79]{hal:towards}
Michael Hallett,
``Towards a Theory of Mathematical Research Programmes (I)'',
{\it British Journal for the Philosophy of Science},
vol.~30 (1979), pp.~1--25.
%
\bibitem[Lakatos 76]{lak:proofs}
Imre Lakatos,
{\it Proofs and Refutations:
The Logic of Mathematical Discovery},
[Cambridge, Cambridge University Press, 1976].
\end{thebibliography}
```

References

[Hallett 79] Michael Hallett, "Towards a Theory of Mathematical Research Programmes (I)", *British Journal for the Philosophy of Science*, vol. 30 (1979), pp. 1–25.

[Lakatos 76] Imre Lakatos, *Proofs and Refutations: The Logic of Mathematical Discovery*, [Cambridge, Cambridge University Press, 1976].

Рис. 3: Необязательный аргумент команды `\bibitem`.

крестные ссылки, надо пропустить LaTeX как минимум дважды: когда LaTeX пропускается первый раз, информация, данная в командных скобках `thebibliography`, записывается в `aux`-файл, а при следующем запуске она используется для получения перекрестных ссылок в выходном результате.

Команда `\cite` имеет такой синтаксис: `\cite[`*text*`]{`*key-list*`}`, где *key-list* — это последовательность из одного или более ключей (если присутствует более одного ключа, они должны разделяться запятыми), а *text* — это необязательная аннотация. Например, входные данные `\cite{lak:proofs,hal:towards}` дают [2, 1], а результат [2, стр.1–8] получен командой `\cite[стр.1--8]{lak:proofs}`.

Команда `\bibitem` может иметь необязательный аргумент, как это показано в верхней части рис.3 на стр.33. Он используется в ссылках в документе в качестве метки. В этом случае "ширина" аргумента командных скобок `thebibliography` должна быть больше необязательного аргумента команды `\bibitem`, используемой внутри этого аргумента. В этом случае `\cite{lak:proofs}` дает [Lakatos 76], а `\cite{hal:towards}` дает [Hallett 79].

13.2 Использование BibTeXa

BibTeX~– это широко доступная система, используемая вместе с LaTeXом для подготовки библиографий. Ее автор — Орен Паташник (Oren Patashnik).[16] Чтобы ее использовать, надо сначала создать файл с расширением `bib`, содержащий последовательность элементов (источ-

[16] Если BibTeX Вам доступен (имеется на Вашем компьютере или в сети), то где-то около него должна быть и документация Паташника — она называется "BibTeXing". В ней значительно больше информации о BibTeXe, чем содержится в этой книге.

ников) — структура их объясняется в следующем подразделе — а затем включить в свой входной файл (в дополнение к стандартным командам `\cite`) специфические команды BibTeXa, указывающие, как форматировать библиографию, которую производит BibTeX, и какие имена `bib`-файлов, содержащих базы данных источников, он должен использовать. Эти команды объясняются в каталоге.

13.2.1 Структура bib-файла

Файл с расширением `bib` содержит один или более элементов, которые выглядят, например, так:

```
@book{il:pr:lmd,
  address   = ``Cambridge'',
  author    = ``Imre Lakatos'',
  isbn      = ``0 521 21078 X (hard covers)
               0 521 29038 4 (paperback)'',
  note      = ``Edited by John Worrall and Elie Zahar'',
  publisher = ``Cambridge University Press'',
  title     = ``Proofs and Refutations:
               The Logic of Mathematical Discovery'',
  year      =  1976}
```

Общий вид такого элемента — это *publication-type*{*key, field-list*}. Возможности для *publication-type* (тип-публикации) приведены на рис.4, стр.35 вместе с кратким описанием того, какого сорта материал требуется для каждого вида элемента. В отличие от LaTeXa, большая часть BibTeXa не различает прописные и строчные буквы. Так, `@book` можно записать как `@BOOK` или даже как `@BooK`. Вместо фигурных скобок можно заключить *key* (ключ) и *field-list* (список полей) в круглые скобки. Если внутри этих ограничителей встречаются любые другие фигурные скобки, то они должны следовать парами — это относится даже к `\{` и `\}`.

Ключ *key* — это то, что надо поставить во всех командах `\cite`, которые используются для ссылок на эту публикацию. Например, чтобы сослаться на книгу Lakatos'a *Proofs and Refutations*, надо использовать команду `\cite{il:pr:lmd}`.

Список полей *field-list* отделяется от ключа запятой и представляет собой список полей, разделенных запятыми. Например, `author = ``Imre Lakatos''`; здесь `author` — это имя поля, а ```Imre Lakatos''` — текст поля. В именах полей прописные и строчные буквы не различаются, поэтому `author` можно записать и как `AUTHOR` или даже как `AutHOr`. Список полей заключается либо в двойные кавычки, либо в фигурные скобки. Если текст поля состоит *только* из цифр, например 1076, то его можно не заключать в кавычки. (В ключе *key* и в списке полей *field-list* многочисленные пробелы вокруг запятых или знаков равенств не влияют на выходной результат.)

Далее следует полный список имен полей, которые в настоящее время распознает BibTeX:

```
address          annote       author
booktitle        chapter      crossref
edition          editor       howpublished
institution      journal      key
month            note         number
organization     pages        publisher
school           series       title
type             volume       year
```

В зависимости от типа публикации (*publication-type*) поля бывают обязательными, необязательными и игнорируемыми. Например, при подготовке элемента даты базы для книги

`@article`{*key,field-list*} Этот вид элемента библиографической базы данных используется для статей, которые были опубликованы в журналах или периодических изданиях.

`@book`{*key,field-list*} Используется для книг с названием издателя.

`@booklet`{*key,field-list*} Используется для трудов, которые напечатаны и переплетены, но на которых нет указания на то, кто их произвел.

`@conference`{*key,field-list*} Используется для статей или документов, которые опубликованы в трудах какой-нибудь конференции. (Этот вид элемента в точности совпадает с типом `@inproceedings`.)

`@inbook`{*key,field-list*} Используется для глав или для других частей книги. Его можно использовать даже для группы страниц книги.

`@incollection`{*key,field-list*} Используется для глав или для других частей книги, которые имеют собственное название. Например, книга, главы которой написаны разными авторами.

`@inproceedings`{*key,field-list*} Используется для статей или для документов, которые опубликованы в трудах какой-либо конференции. (Этот вид элемента в точности совпадает с типом `@conference`.)

`@manual`{*key,field-list*} Используется для руководств или для других видов технической документации.

`@mastersthesis`{*key,field-list*} Используется для диссертации на соискание степени магистра.

`@misc`{*key,field-list*} Используется для тех вещей, на которые Вы хотите сослаться, но которые не подходят к чему-либо еще.

`@phdthesis`{*key,field-list*} Используется для докторских диссертаций.

`@proceedings`{*key,field-list*} Используется для трудов конференции в отличие от отдельного документа из этих трудов.

`@techreport`{*key,field-list*} Используется для научных или для технических отчетов отделений учебных заведений или исследовательских лабораторий.

`@unpublished`{*key,field-list*} Используется для документов, например, машинописных, которые имеют название и автора, но нигде не были изданы. К этой же категории относятся документы самиздата.

Рис. 4: Различные виды типов публикаций *publication-type*.

типа Lakatos *Proofs and Refutations*, должны присутствовать только поля `publisher, title` и `year`. К тому же должны присутствовать либо поле `author`, либо поле `editor` (но не оба вместе) и может присутствовать, самое большее, одно из полей `volume` или `number` (могут оба отсутствовать). Поля `address`, `crossref`, `edition`, `key, month`, `note` и `series` являются необязательными, но используются, если присутствуют. Поля всех других видов просто игнорируются. На рис.5, стр.37 показано шесть различных типов полей. Они обозначены пятью буквами `A, E, O, R, V` и пробелом. Эти типы полей имеют следующий смысл:

`A` Используется в колонке `@inbook` для обозначения, что в аргумент *field-list* можно включать либо поле `chapter`, либо поле `pages`, либо оба вместе. Если отсутствуют оба эти поля, выдается предупреждающее сообщение, но результат выглядит правильно.

`E` Используется в колонке `@book` и `@inbook` для указания, что может быть включено либо поле `author`, либо поле `editor`. Если присутствуют оба эти поля, то при запуске BibTeXа выдается предупреждающее сообщение и поле `editor` игнорируется. Если отсутствуют оба эти поля — а также отсутствует поле `key` — Вам выдается предупреждающее сообщение о том, что BibTeX не может отсортировать этот элемент библиографии. Однако в библиографии он все же появится.

`O` Используется для указания, что данное поле является необязательным. Информация, если она присутствует, будет использоваться, но ее отсутствие не вызовет никаких проблем.

`R` Используется для указания, что данное поле является обязательным. Если поле отсутствует, то при запуске BibTeXа будет выдано сообщение об ошибке и, возможно, полученная библиография не будет должным образом сформатирована.

`V` Эта буква встречается в колонках для следующих типов элементов библиографии `@book, @conference, @inbook, @incollection, @inproceedings` и `@proceedings`. Она используется для указания, что может присутствовать либо поле `volume`, либо поле `number`. Если оба эти поля присутствуют, выдается сообщение об ошибке и поле `number` игнорируется. Если присутствует поле `volume` либо оба поля отсутствуют, проблем не возникает. Если присутствует поле `number` при отсутствии поля `series`, Вы получаете предупреждающее сообщение, но результат получается правильным.

␣ Указывает, что поле, даже если оно присутствует, игнорируется.

13.2.2 Подготовка библиографии

Для того чтобы BibTeX получил для Вас библиографию, Вам необходимо внутри командных скобок `document` Вашего входного файла поставить команды `\bibliographystyle` и `\bibliography`. Команда `\bibliographystyle` обычно следует сразу же за командой `\begin{document}` и обязательно должна стоять до всех команд `\cite`. Команда `\bibliography` обычно располагается ближе к концу документа в том месте, где Вы хотите, чтобы появилась полученная библиография. Если Вы хотите поместить в библиографию работу, которую Вы не цитируете — но которая имеется в Вашем `bib`-файле — надо использовать команду `\nocite{`*key*`}`. Команда `\nocite{*}` во входном файле указывает на то, что *каждая* работа из базы данных BibTeXа, упомянутая в команде `\bibliography`, должна появиться в полученной библиографии.

Синтаксис команды `\bibliography` таков:

`\bibliography{`*bib-file-list*`}`

	@article	@book	@booklet	@conference	@inbook	@incollection	@inproceedings	@manual	@mastersthesis	@misc	@phdthesis	@proceedings	@techreport	@unpublished
address		O	O	O	O	O	O	O	O		O	O	O	
annote														
author	R	E	O	R	E	R	R	O	R	O	R		R	R
booktitle				R		R	R							
chapter					A	O								
crossref	O	O	O	O	O	O	O	O	O	O	O	O	O	O
edition		O			O	O		O						
editor		E		O	E	O	O					O		
howpublished			O							O				
institution													R	
journal	R													
key	O	O	O	O	O	O	O	O	O	O	O	O	O	O
month	O	O	O	O	O	O	O	O	O	O	O	O	O	O
note	O	O	O	O	O	O	O	O	O	O	O	O	O	R
number	O	V		V	V	V	V					V	O	
organization				O			O	O				O		
pages	O			O	A	O	O							
publisher		R		O	R	R	O					O		
school									R		R			
series		O		O	O	O	O					O		
title	R	R	R	R	R	R	R	R	R	O	R	R	R	R
type					O				O		O		O	
volume	O	V		V	V	V	V					V		
year	R	R	O	R	R	R	R	O	R	O	R	R	R	O

Рис. 5: Статус полей элемента базы данных BibTeX для различных типов публикаций.

где *bib-file-list* — это список первых или основных имен одного или более `bib`-файлов. Если в списке более одного члена, они должны разделяться запятыми.

Команда `\bibliographystyle` имеет следующий синтаксис:

`\bibliography`{*bib-style*}

где стандартными возможностями для стиля *bib-style* являются `plain, unsrt, abbrv` и `alpha`. В стиле библиографии, полученной опцией `plain`, элементы библиографии автоматически отсортированы в алфавитном порядке по фамилии автора, работы одного автора — по возрастанию года, а работы одного автора, изданные в одном и том же году — по названию.

Если вместо `plain` выбрана опция `unsrt`, элементы полученной библиографии будут выглядеть точно так же, как и при опции `plain`, но не будут отсортированы. Они появятся в библиографии в том порядке, в каком на них делались ссылки во входном файле.

Если выбрана опция `abbrv`, то порядок элементов в полученной библиографии будет таким же, как при опции `plain`, но имена и некоторые другие слова будут приведены с сокращениями.

Если выбрана опция `alpha`, библиография будет отсортирована так же, как при опции `plain`, но будет выглядеть несколько иначе.

Чтобы LaTeX и BibTeX создал и включил библиографию в выходной результат, соответствующий Вашему входному файлу, Вам надо запустить и LaTeX и BibTeX с этим файлом: прежде всего Вы обычным образом обработаете свой входной файл LaTeXом, а затем BibTeXом (во многих системах это делается командой `bibtex` *file*, где *file* — это имя Вашего входного файла) а затем дважды запустите на этом файле LaTeX. (В исключительных обстоятельствах Вам придется обработать свой входной файл даже большее число раз.)

13.3 Получение алфавитного указателя

Алфавитный указатель в LaTeXе получается командными скобками `theindex`. Внутри этих командных скобок можно использовать команды `\item`, `\subitem`, `\subsubitem` и `\indexspace`. Команда `\item` начинает элемент получаемого алфавитного указателя, команда `\subitem` начинает его подэлемент, который набирается с небольшим отступом, а отступ того, что производит команда `\subsubitem`, делается еще бóльшим. Для того чтобы отделить группы элементов, начинающиеся с разных букв алфавита, используется команда `\indexspace`.

14 Подготовка “глоссария”

В LaTeXе имеются команды `\makeglossary` и `\glossary`, которые работают аналогично командам `\makeindex` и `\index`. Записывается (или перезаписывается) файл с расширением `glo`, содержащий команды `\glossaryentry`, которые аналогичны командам `\indexentry`. Заметим, что командных скобок `theglossary`, аналогичных командным скобкам `theindex`, не существует. Для получения того, что многие люди считают глоссарием, лучше всего использовать командные скобки `description`, которые объясняются в каталоге.

15 Использование команд Plain TeXa

LaTeX реализуется как “макропакет” TeXa в виде ряда заранее определенных TeXовских команд. Plain TeX — это стандартная версия TeXa, состоящая из “сырого” TeXa плюс макропакет `plain`. Большинство команд plain TeXa можно использовать в LaTeXе, но делать это нужно осторожно. LaTeX разработан таким образом, что его команды образуют единую систему. Пришлось пойти на многие компромиссы, чтобы обеспечить правильную работу

какой-либо команды, когда она разумно используется совместно с другими LaTeXовскими командами. LaTeXовская команда может работать неправильно, если она используется вместе с командами plain TeXa, не описанными в этой книге.

Заранее трудно сказать, будут ли проблемы с какой-либо командой plain TeXa, это выясняется только на практике. Общее правило состоит в том, чтобы не сочетать LaTeXовскую команду или командные скобки с командами plain TeXa, которые могут модифицировать параметры. Так, нельзя использовать команду plain TeXa, например `\hangindent`, которая модифицирует параметры TeXa, создающие абзацы внутри каких-либо LaTeXовских командных скобок, создающих списки.

Не следует модифицировать какие-либо параметры, используемые программой LaTeXa `\output`, способом, не указанным в этой книге. В частности, вам следует забыть о бóльшей части главы 15 из книги Дональда Кнута "Все про TeX"[1]. Однако LaTeX подчиняется всем TeXовским соглашениям о распределении регистров, так что вы можете определять свои собственные счетчики, боксы и т.п. обычными командами TeXa.

Ниже перечислены все команды plain TeXa, определения которых были изъяты или изменены в TeXe. Здесь не перечислены LaTeXовские команды, которые близки соответствующим версиям plain TeXa, а также некоторые "внутренние" команды, имена которых содержат символы `@`.

15.1 Команды табулирования

Командные скобки LaTeXa `tabbing` выводят из употребления следующие команды:

```
\tabs       \tabsdone     \settabs      \+
\tabset     \cleartabs    \tabalign
```

15.2 Вывод, сноски и рисунки

Следующие команды, требующие программы вывода plain TeXa, не действуют. Они заменяются LaTeXовскими командами, создающими сноски, и его командными скобками `figure` и `table`.

```
\pageno       \nopagenumbers   \makeheadline   \footstrut
\headline     \advancepageno   \makefootline   \topins
\footline     \nopagenumbers   \dosupereject   \topinsert
\normalbottom \plainoutput     \pagecontents   \midinsert
\folio        \pagebody        \vfootnote      \pageinsert
                                               \endinsert
```

15.3 Команды выбора шрифта

Следующие команды plain TeXa не определены в LaTeXe. Найдите файл `lfonts.tex` и в нем соответствующие LaTeXовские команды:

```
\fivei        \fivebf         \sevensy
\fiverm       \seveni         \teni
\fivesy       \sevenbf        \oldstyle
```

Посмотрите определение `\oldstyle` в plain TeXe и разберитесь, как получить такие результаты в LaTeXe.

15.4 Выравнивание уравнений

Следующие команды plain TEXa выводятся из употребления LATEXовскими командными скобками `eqnarray` и `eqnarray*`:

```
\eqalign        \eqalignno        \leqalignno
```

15.5 Разное

TEXовская команда `\beginsection` заменяется LATEXовскими командами рубрикации; `\end` и `\bye` заменяются на `\end{document}`. Имена команд `\centering` и `\line` из plain TEXa были присвоены LATEXовскими командами. Большинство функций, которые выполняла команда `\line` в plain TEXe, могут выполняться командными скобками `center`, `flushleft` и `flushright`. Команда `\magnification` в plain TEXe не имеет аналога в LATEXe. Увеличение выходных данных часто можно осуществлять с помощью программы, которая печатает `dvi`-файлы.

16 Работа над ошибками

Одна из наиболее распространенных ошибок начинающих пользователей LATEXa — это неправильный ввод имен некоторых команд. Например, если Вы в некотором входном файле `file.tex` вместо команды `\maketitle` поставите `\maketitel`, а затем запустите LATEX, то обработка Вашего файла прервется, а на терминале появится что-то вроде:

```
This is TeX, C Version 3.14t3
(file.tex
LaTeX Version 2.09 <7 Dec 1989>
(/bham/tex/inputs/article.sty
Document Style `article' <16 Mar 88>.
(/bham/tex/inputs/art12.sty)) (file.aux)
! Underfined control sequence.
1.5 \maketitel

?
```

(А может быть, и не в точности это, поскольку некоторая часть информации связана с особенностями системы, в которой Вы используете LATEX, в частности, имена пути файлов `article.sty` и `art12.sty`). Если в ответ на подсказку Вы введете строчную букву `e`, то окажетесь в текстовом редакторе, причем курсор будет находиться в начале строки, в которой встретилось ошибочное имя команды. Можно в ответ на подсказку ввести букву `h`, и тогда LATEX или TEX обычно даст дополнительную информацию о том, что идет неправильно. Иногда у Вас могут возникнуть затруднения с выходом из LATEXa; в этом случае можно остановить работу, ответив на подсказку `I\stop`. Вы можете получить много разных сообщений об ошибках, но большинство из них сами себя объясняют, поэтому здесь их объяснять не надо.

Что еще часто встречается при обработке файлов LATEXом — это предупреждающее сообщение `Overfull \hbox`. Это предупреждающее сообщение означает, что некий текст залезает на правое поле страницы. Лучший способ решить такую проблему — это переписать предложение, которое ее порождает. Альтернативный вариант — явно указать TEXу, где он может сделать перенос. Для разрешения этой проблемы можно также попробовать использовать команды `\hyphenate`, `\sloppy` и `\linebreak`. Все они объяснены в каталоге.

В процессе работы можно получать информацию о командах и параметрах, используя примитивные команды TEXa `\show` (чтобы посмотреть на определение макрокоманды) и `\showthe`

(чтобы узнать значение внутреннего параметра). Так, например, если Вы введете `\show\bar`, на экране появится

```
> \bar=macro
->\mathaccent "7016
\show\bar
?
```

а после `\show\hsize` — только

```
> \hsize=\hsize
\show\hsize
?
```

потому что `\hsize` — это примитив. Зато после команды `\showthe\hsize` Вы увидите, например

```
> 469.75499 pt
\showthe\hsize
?
```

т.е. текущее значение примитива `\hsize`. После `?` работу можно продолжить нажатием клавиши CR. Имеются и другие `\show`-команды, которые помогут Вам заглянуть в глубины TEXa.

Часть II
Каталог команд LaTeXa

Список литературы

[1] Дональд Кнут. *Все про TEX*, перевод с англ. — РДТЕХ, 1993.

[2] Leslie Lamport. LaTeX. *A Document Preparation System.* — Addison Wesley, 1985.

[3] Antony Diller. LaTeX. *Line by Line.* — John Wiley & Sons, 1993.

[4] Грицаенко И.А., Клименко С.В. LaTeX. *Руководство для пользователя*: Препринт ИФВЭ 97–55. — Протвино, 1997.

[5] Лисина М.В. *Plain TEX. Основные понятия и каталог команд*: Препринт ИФВЭ 95–58. — Протвино, 1995.

[6] Бердников А.С., Туртиа С.Б. *TEX и Графика.* — Политехника: Санкт-Петербург, 1995.

Рукопись поступила 22 января 1998 г.