

ИНСТИТУТ ПРОБЛЕМ МЕХАНИКИ
РОССИЙСКОЙ АКАДЕМИИ НАУК

А. И. Журов

ОСНОВЫ ТрХа

Препринт № 518-2

Москва 1997 год

Аннотация

Рассматривается компьютеризированная наборная система \TeX , позволяющая получать документы типографского качества. Система дает возможность также существенно сокращать время выхода в свет публикаций за счет предоставления издательству качественного оригинал-макета для прямого воспроизведения. Дается описание основных возможностей и понятий \TeX 'а, знание которых необходимо для начального изучения \TeX 'а. По возможности используется принятая в нашей стране полиграфическая терминология. Оригинал-макет препринта изготовлен средствами \TeX 'а с помощью пакета $\mu\text{\TeX}$.

В препринт внесены исправления по сравнению с первоначальным вариантом 1992 года [7].

Abstract

The computerized typesetting system \TeX (by D. Knuth) is considered that allows making documents of typographic quality. The system may significantly reduce the time needed for a book to be published. This may be achieved at the expense of submitting a high-quality camera-ready manuscript to a publisher. In the preprint, \TeX 's fundamental features and notions are considered that are very important to a \TeX novice. When possible, equivalent Russian typesetting terms are used. The camera-ready manuscript of the preprint was prepared by means of \TeX with $\mu\text{\TeX}$, macros collection developed by the author.

Введение

С быстрым развитием компьютерной техники стремительно развиваются и компьютерные технологии во многих областях человеческой деятельности, в том числе и в науке. Ученых компьютеры интересуют главным образом как инструмент для получения, проверки, обработки и представления в удобной для восприятия форме результатов исследовательской работы. Всем, занимающимся исследовательской деятельностью, известно скольких усилий стоит подготовка публикаций научных работ.

Еще недавно, а порой и до сих пор, рукопись отдавали машинистке, та печатала несколько экземпляров текста, а формулы приходилось вписывать вручную. В таком виде работа отдавалась для публикации в издательство. Там производился типографский набор, коррекция и т.п. В результате с момента изготовления машинописной рукописи до появления ее в печати проходило не менее года, а зачастую и больше. Если же позже рукопись приходилось дополнять и видоизменять, то почти вся процедура повторялась с самого начала.

Скорость выхода рукописи в свет можно увеличить за счет представления издательству качественного *оригинал-макета для прямого воспроизведения*¹.

Используя компьютер и хороший принтер, можно получить высококачественный макет рукописи с помощью средств, предоставляемых различными текстовыми процессорами, настольными издательскими системами. Среди них особое место занимает система $\text{T}_{\text{E}}\text{X}^2$, специально предназначенная для изготовления отличных документов, в которых часто встречаются математические формулы. $\text{T}_{\text{E}}\text{X}$ — компьютеризированная наборная система, автором которой является Дональд Кнут (Donald E. Knuth). Документы, приготовленные средствами $\text{T}_{\text{E}}\text{X}$ 'а, удовлетворяют самым высоким мировым типографским стандартам. Многие западные издательства, имеющие дело с научными публикациями, предпочитают работать с файлами в формате $\text{T}_{\text{E}}\text{X}$ 'а.

¹ **оригинал-макет для прямого воспроизведения** — завершенные по всем элементам страницы будущего издания, предназначенные для перевода (по той или иной технологии) в печатную форму

² название происходит от греческого корня $\tau\epsilon\chi$, означающего 'искусство' и 'технология', и произносится 'тех'

Авторы пересылают подготовленные публикации в виде файлов на дискетах или по электронной почте.

Конечно, чтобы создать рукопись в виде Т_ЕX’овского файла нужно затратить больше усилий по сравнению с печатью на пишущей машинке и ручным вписыванием формул. Однако, это с лихвой окупается качеством документа (напечатанного даже на обычном матричном принтере) и, что более важно, возможностью легко видоизменять документ, внося небольшие изменения в файл. Более того, можно легко изменять формат всего документа (размер страниц, размер шрифта, вид заголовков и т.п.), исправив всего несколько команд в начале входного файла.

Вообще говоря, один из основных критериев оценки компьютерных систем подготовки документации — время, затрачиваемое на (видо)изменение формата документа. Здесь проявляется одно из основных достоинств Т_ЕX’а.

Выходной Т_ЕX’овский файл включает как текст, так и команды, определяющие как общий вид документа, так и конкретные действия над текстом, формулами, вплоть до отдельного символа. Все команды Т_ЕX’а делятся на две большие группы: *примитивы* и *макросы*¹ (команды, составленные из примитивов и других макросов).

Существует три наиболее известных диалекта Т_ЕX’а: plain Т_ЕX [1] (см. также [4—6]), Л^AТ_ЕX [2] и АМ_S-Т_ЕX [3]². Все они имеют одно и то же ядро, состоящее из примитивов (около 300), а различия заключаются в наборе макросов (более 600). Почти все команды plain Т_ЕX’а доступны из Л^AТ_ЕX’а и АМ_S-Т_ЕX’а. Обычно, когда речь идет о Т_ЕX’е, то имеют в виду plain Т_ЕX.

Л^AТ_ЕX очень удобен, когда имеется *стиль* (дополнительный набор макросов), определяющих общий вид вашего документа. Имеется четыре стандартных стиля: article, book, report, letter. Используя эти стили, пользователю не нужно сосредотачиваться на том, как должен выглядеть его документ, — об этом позаботится Л^AТ_ЕX. Однако для русскоязычной литературы эти стили не вполне подходят. Неболь-

¹ строго говоря, такое деление не совсем точно, поскольку есть команды и других типов

² plain Т_ЕX создан Д. Кнудом; автор Л^AТ_ЕX’а — Л. Лампорт (Leslie Lamport); АМ_S-Т_ЕX создан в Американском математическом сообществе (American Mathematical Society)

шие отклонения от стиля как правило делать несложно, однако в ряде случаев при этом возникают существенные проблемы, для решения которых нужно создавать свой стиль при отсутствии подходящего.

$\mathcal{A}\mathcal{M}\mathcal{S}$ - TEX предназначен специально для математиков и тех, в чьих работах используется много математических формул с большим разнообразием символов.

По мнению автора настоящего препринта лучше всего начинать с plain TEX 'а. Перейти при необходимости на $\text{L}\text{A}\text{T}\text{E}\text{X}$ или $\mathcal{A}\mathcal{M}\mathcal{S}$ - TEX будет несложно.

В препринте излагаются основы TEX 'а. Наиболее полное и, пожалуй, лучшее описание TEX 'а можно найти в [1]. Это весьма большая книга (почти 400 страниц), однако большинству пользователей TEX 'а не понадобится такой объем информации. Чтобы быстрее научиться работать с TEX 'ом необходимо прежде всего хорошо освоить основные правила, понятия, команды TEX 'а. Автор препринта надеется, что препринт поможет в этом не только начинающим, но и тем, кто уже работал с TEX 'ом, а также будет полезен пользователям $\text{L}\text{A}\text{T}\text{E}\text{X}$ 'а и $\mathcal{A}\mathcal{M}\mathcal{S}$ - TEX 'а. В препринте автор старался придерживаться, по возможности, типографско-издательской терминологии [8], принятой для русскоязычной литературы.

В приложении приводятся наиболее употребительные команды TEX 'а.

Оригинал-макет настоящего препринта изготовлен на TEX 'е с помощью разработанного автором пакета $\mu\text{T}\text{E}\text{X}$.

§ 1. Предварительные замечания

Система Т_ЕX включает следующие компоненты¹:

- компилятор (создает из `tex`-файлов `dvi`-файлы);
- формат(ы) (например, файл `plain.fmt`);
- файлы, содержащие необходимые данные о шрифтах (`*.tfm`);
- файлы шрифтов (`*.pk`, `*.gf`, `*.ppl`);
- вьюер (программа просмотра документа на экране);
- программы вывода документа на печатающие устройства;
- различные вспомогательные программы и файлы.

Процедура получения оригинал-макета рукописи состоит из следующих шагов:

- 1) приготовить с помощью любого текстового редактора документ в виде ASCII-файла² с расширением `.tex` или исправить имеющийся файл (далее такой файл будем называть *входным*);
- 2) создать с помощью Т_ЕX-компилятора (программа `tex.exe`) `dvi`-файл (в независимом от выходного устройства формате — `device independent`) с расширением `.dvi`; при возникновении ошибок можно перейти к шагу 1;
- 3) просмотреть `DVI`-файл на экране графического дисплея с помощью вьюера (программа `dviscr.exe`); при обнаружении ошибок можно вернуться к шагу 1;
- 4) создать из `DVI`-файла файл в формате конкретного выходного устройства (программа `dvihplj.exe` — для лазерного принтера, `dvidot.exe` — для матричного принтера, `dvips.exe` — для принтеров семейства PostScript);

¹ названия файлов приводятся в соответствии с имеющейся у автора русифицированной версией Т_ЕX'а для IBM-совместимого персонального компьютера с операционной системой MS DOS или PC DOS

² предполагается возможность использования следующих ASCII-символов:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789,;:!"'/*@+-=<>() []_
\ { } $ & # ^ _ % ~ |
АБВГДЕЖЗИЙКЛМНОПРТУФХЦЧШЩЪЫЬЭЮЯ
абвгдежзийклмнопртуфхцчшщъыьэюя
```

(Знаком `_` здесь и далее в препринте обозначается пробел.)

- 5) распечатать полученный на шаге 4 файл (командой ‘`copy/b <имя файла> prn`’ из командной строки DOS).

§ 2. Элементарные представления о Т_EX’е

Рассмотрим пример. Как правило, текст набирают без учета расположения отдельных слов и других деталей. Т_EX сам позаботится об этом:

Это первый абзац небольшого документа на 400 знаков, полученного с помощью Т_EX’а. Строки абзаца выравниваются автоматически по левому и правому краям.

Для этого Т_EX увеличивает или уменьшает пробелы между словами и/или переносит часть слова, не уместившегося в строке. (Однако он не может исправлять ошибки.)

Пустая строка во входном файле означает начало нового абзаца!

В результате обработки Т_EX’ом входного файла получится следующий документ:

Это первый абзац небольшого документа на 400 знаков, полученного с помощью Т_EX’а. Строки абзаца выравниваются автоматически по левому и правому краям. Для этого Т_EX увеличивает или уменьшает пробелы между словами и/или переносит часть слова, не уместившегося в строке. (Однако он не может исправлять ошибки.)

Пустая строка во входном файле означает начало нового абзаца!

Отметим, что строки входного файла могут быть произвольной длины. Т_EX автоматически форматирует абзац: делает абзацный отступ и выравнивает строки. Число пробелов между словами не играет для Т_EX’а роли — он просто игнорирует все пробелы кроме первого. В конце каждой строки Т_EX вставляет пробел вместо знака перевода

строки `\return` (получаемого нажатием клавиши `[Enter]`), поэтому все пробелы в начале следующей строки игнорируются, как это сделано в строке

```
      \dots\Для этого TeX
```

(То же самое относится и к пробелам, полученным нажатием клавиши `[Tab]`.)

После таких *знаков прерывания* как точка, двоеточие, вопросительный и восклицательный знаки TeX слегка увеличивает пробел по сравнению с обычными пробелами между словами.

С другой стороны пробел между символом `'` и словом `'Однако'` в рассматриваемом примере — грубая ошибка, которая приводит к ненужному пробелу на печати. TeX посчитал `'` словом из одной буквы, а поскольку оно пришлось на конец строки, то получилось, что скобка и слово `'Однако'` оказались оторванными друг от друга.

Признаком начала нового абзаца является пустая строка. При этом для TeX'a не важно, сколько пустых строк идет подряд: он игнорирует все после первой аналогично тому, как он поступает с пробелами.

Рассмотрим более сложный пример. Пусть имеется следующий входной файл

```
\hsize=11.3cm \vsize=15.6 cm
\hrule
\vskip 1mm
Один \TeX ник как-то сказал: ‘‘При наборе
текста различайте дефис ‘-’, тире (‘---’)
и полутире (‘--’), а также знак минуса ‘$-$’.
\TeX\ знает, что знак процента % всегда
означает комментарий до конца строки.
Если вы захотите написать какое-
либо слово на двух строках, то знак про%
цента вам поможет. Вместо многоточия ... следует
набирать \dots \par Необходимо ставить знак ударения
в таких словах, как ‘з\’амок’ и ‘зам\’ок’.
Видно, что работа с \TeX’ом~---
вес\’елое занятие.’’
```



```
\vskip 0.1cm \hrule
\bye
```

В результате вы получите следующее:

Один Т_ЕXник как-то сказал: “При наборе текста различайте дефис ‘-’, тире (‘—’) и полутире (‘-’), а также знак минуса ‘-’. Т_ЕX знает, что знак процента означает комментарий до конца строки. Если вы захотите написать какое-либо слово на двух строках, то знак процента вам поможет. Вместо многоточия ... следует набирать ...

Необходимо ставить знак ударения в таких словах, как ‘за́мок’ и ‘замо́к’. Видно, что работа с Т_ЕX’ом — весёлое занятие.”

Всё, что начинается с ‘\’ — это *управляющие последовательности*¹ Т_ЕX’а. Они бывают двух типов: *управляющие слова* (control word) и *управляющие символы* (control symbol). В первом случае за символом ‘\’ следует одна или несколько букв (a...z, A...Z), например, \input, \P, \alpha, \infty и т.п. Следует иметь в виду, что

- Т_ЕX узнаёт об окончании управляющего слова по первой встретившейся небукве;
- строчные и прописные буквы считаются различными (например, \TeX и \tex — два разных управляющих слова);
- все пробелы после управляющего слова игнорируются.

Во втором случае за символом ‘\’ следует любой символ-небуква, например, \\$, \%, _, \1, \; и т.п. Пробел после управляющего символа не игнорируется. Исключение составляет _, т.е. _ _ и _ _ дадут одинаковый результат.

В рассмотренном примере управляющие слова — \hsize, \vsize, \hrule, \vskip, \TeX, \dots, \par и \bye, а управляющие символы — _, \" и \’.

Команды \hsize и \vsize задают соответственно горизонтальный и вертикальный размеры текста на странице. Кстати, в препринте используются именно указанные в примере размеры (11,3 мм и 15,6 мм).

Команда \hrule проводит горизонтальную линию на всю ширину страницы (заданной командой \hsize), а команда ‘\vskip 1mm’ задает пробел в 1 мм по вертикали.

¹ в дальнейшем чаще будет использоваться термин *команды*

Команда `\TeX` определяет слово `TeX`. Заметим, что `\TeX_ник` дает `TeXник`, но было бы ошибкой набрать `\TeXник`.

Команда `_` задает пробел `_`, который `TeX` не игнорирует. Поэтому `\TeX_знает` дает `TeX знает`, а `\TeX_знает` дает `TeXзнает`.

При печати на машинке знак минуса, дефис и тире обозначаются одним и тем же символом, однако это не допускается при изготовлении документов книжного качества. В составных словах типа ‘как-то’, ‘где-нибудь’, ‘научно-технический’ и т.д., используется один знак ‘-’. Чтобы получить тире, следует набрать ‘---’. Полутире ‘--’ применяется для обозначения диапазона изменения чисел, например, ‘стр. 12–26’, или в контексте типа ‘Рис. А–2’¹. Знак минуса используется в формулах, например, $a = c - b$ (формулы набираются в математическом режиме, который включается и выключается символом ‘\$’: `$a=b-c$`).

Чтобы получить двойные кавычки “...”, следует набрать два раза одинарные: ‘...’. С одинарными кавычками всё ясно без объяснений².

Следует отметить, что некоторые комбинации латинских букв `TeX` трактует как единое целое: буквосочетания ‘ff’, ‘fi’, ‘fl’, ‘ffi’ и ‘ffl’ дают соответственно ‘ff’, ‘fi’, ‘fl’, ‘ffi’ и ‘ffl’. Такие буквосочетания называются *лигатурами* (ligature). Кстати к лигатурам относятся и ‘---’, ‘--’, ‘’’, ‘‘’ (см. также приложение). `TeX` также делает автоматический *кернинг* (kerning), т.е. сближение некоторых букв. Например, буквосочетание ‘AV’ дает ‘AV’, без кернинга было бы ‘AV’.

Знак процента ‘%’ говорит `TeX`у, что оставшуюся часть строки входного файла нужно игнорировать. Это свойство полезно не только для вставки комментариев, но и в случае, когда нужно отменить действие символа `\return` в конце строки. Например, поскольку в файле на самом деле набрано ‘**какое-`\return`либо**’, то на печати получилось ‘**какое- либо**’ (`\return` был заменен на пробел). Но ‘**про`\return`цента**’ дает ‘процента’.

Если нужно набрать в тексте многоточие, то используется команда `\dots` (дает ...), но не ... (дает ...).

¹ в русскоязычной литературе вместо полутире обычно используют тире: ‘стр. 12—26’

² вместо “...” и ‘...’ в русскоязычной литературе обычно используют «...», „...” или „...”

Выше уже говорилось, что признаком начала нового абзаца во входном файле является пустая строка. Однако вместо пустой строки можно поставить команду `\par`. Заметим, что любая последовательность пустых строк и команд `\par` интерпретируется Т_ЕX'ом как одна команда `\par`.

Во многих языках для написания слов используются *акценты* (знаки ударения). Например, русские слова 'весёлый' и 'замók' набраны с помощью акцентов `\" и \'`: 'вес\"елый' и 'зам\'ок'. Полный перечень команд Т_ЕX'а, определяющих акцентирование букв в тексте, приводится в приложении.

Т_ЕX автоматически разбивает абзац на строки по пробелам между словами или же при необходимости делает перенос слов. Но, если два (или более) соседних слова логически тесно связаны (например, 'Глава 1', 'стр. 8', 'переменная *y*' и т.п.), то нужно сообщить Т_ЕX'у, что разрыв между ними запрещен. Это можно сделать с помощью символа '~' (тильда), который означает пробел, но запрещает разрыв строки в этом месте.

Обычно не принято, чтобы строка начиналась с тире. Поэтому в рассматриваемом выше входном файле было набрано '`... работа с \TeX'ом--- ...`'. Ниже приводится ряд примеров, показывающих, как использовать символ '~':

параграф~\S~1	параграф § 1
Приложение~А	Приложение А
рис.~3	рис. 3
теорема~1.2	теорема 1.2
А.~Соловьев	А. Соловьев
Donald~Е. Knuth	Donald E. Knuth
функция~\$f(x)\$	функция $f(x)$
1,~2 или~3	1, 2 или 3
от 0 до~1	от 0 до 1
равно~15	равно 15
равно 15~см	равно 15 см

§ 3. Шрифты

Под *шрифтом* понимается комплект *литер*¹, необходимых для воспроизведения букв определенного алфавита, знаков и цифр. Шрифты различают по *рисунку*, *начертанию* (прямой, наклонный, курсивный), *насыщенности* (светлый, полужирный, жирный) и *размерам*.

Обычный текст в различного рода публикациях набирают *прямым светлым* (normal roman) шрифтом, каким набран этот препринт. Однако иногда возникает необходимость выделить часть текста **полужирным** или *курсивом*. Т_ЕX умеет работать с различными наборами (шрифтами), содержащими до 256 литер. Переключение с одного шрифта на другой осуществляется с помощью управляющих последовательностей. Например, конец второго предложения можно набрать так:

```
часть текста \bf полужирным \rm или \it курсивом. \rm
```

В plain Т_ЕX'e определены следующие команды переключения шрифта:

<code>\rm</code>	прямой светлый	normal roman
<code>\bf</code>	полужирный	boldface
<code>\it</code>	<i>курсив</i>	<i>italic</i>
<code>\sl</code>	<i>наклонный</i>	<i>slanted</i>
<code>\tt</code>	равноширинный	typewriter

В начале документа по умолчанию действует прямой шрифт (`\rm`).

Отметим, что курсив и наклонный шрифт отличаются друг от друга, хотя оба имеют наклон. Курсив удобно использовать для *выделения слов* в прямом шрифте. *Наклонный шрифт* можно также применять для *выделения* наряду с курсивом².

Поскольку, как правило, основная часть текста набирается прямым шрифтом, то не очень удобно писать ‘`\rm`’ всякий раз, когда

¹ в препринте используются следующие термины: **литера** — элемент шрифтового набора, используемый для построения изображения того или иного печатного знака; **знак** — изображение на печати, получаемое с помощью одной или нескольких литер; **символ** — любой ASCII-код или управляющая последовательность во входном файле

² например, принятые у нас типографско-издательские термины вводятся в препринте наклонным шрифтом, а термины, которые относятся непосредственно к Т_ЕX'у, и другие — курсивом

нужно вернуться к прямому шрифту. Однако \TeX обладает способностью локализовывать действие команд (в частности команд переключения шрифта) внутри фигурных скобок. Например, приведенную выше строку можно набрать и так:

часть текста `{\bf полужирным}` или `{\it курсивом}`.

Оба указанных способа смены шрифта работают одинаково хорошо. Однако предпочтение лучше отдавать последнему.

Курсивный и *наклонный* шрифты имеют наклон, поэтому при переходе к прямому шрифту кажется, что пробел в месте перехода меньше обычного. Чтобы слегка увеличить этот пробел нужно использовать команду ‘\’ непосредственно перед переключением на прямой шрифт. Сравните две строки:

`{\it курсивный}` и `{\sl наклонный}` шрифты
`{\it курсивный\}` и `{\sl наклонный\}` шрифты

что дает

курсивный и *наклонный* шрифты
курсивный и *наклонный* шрифты

Управляющий символ ‘\’ сообщает \TeX у, что нужно вставить *поправку на наклон* (italic correction) для предыдущей буквы. Эта поправка для различных букв не одинакова, но \TeX об этом знает и не ошибется.

Иногда, правда, поправка на наклон не нужна. Есть общее правило: при переключении со шрифта, имеющего наклон, на прямой шрифт необходимо использовать ‘\’, если только следующий символ не точка или запятая. Перед другими знаками препинания желательно вставлять поправку на наклон. Например,

поправка на `{\it наклон}`. поправка на *наклон*.
простое `{\it правило\}`: простое *правило*:

Размер шрифта, или *кегель*, измеряется в *пунктах*¹. Например, размер шрифта, которым набран основной текст — 10 пунктов (длина

¹ в \TeX 'е 1 пункт = 1/72,27 дюйма \approx 0,351 мм, однако следует иметь в виду, что в нашей стране (и в Европе) принято 1 пункт = 1/2660 м \approx 0,376 мм

тире и высота круглых скобок в точности равны 10 пунктам). Заголовки в параграфов препринте набраны шрифтом с кеглем 12 пунктов, сноски — 8 пунктов, индексы в формулах — 7 пунктов, индексы второго порядка — 5 пунктов. Шрифты кеглей 6 и 9 приводятся ниже для примера. По кеглям шрифты имеют свои названия¹. Например:

Пять пунктов	перл
Шесть пунктов	нонпарель
Семь пунктов	миньон
Восемь пунктов	петит
Девять пунктов	боргес
Десять пунктов	корпус
Двенадцать пунктов	цицero
Четырнадцать пунктов	миттель

В plain TeX'e определены команды переключения на шрифты кеглей 5, 7 и 10 пунктов. Имена соответствующих команд состоят из английского числительного (five, seven, ten) и имени шрифта (rm, bf, sl, it, tt), например, `\fiverm`, `\sevembf`, `\tensl`. Команды `\rm`, `\bf` и т.д. обычно соответствуют шрифтам размера 10 пунктов. Однако в сложных формулах они могут определять размер 7 или 5 пунктов.

Для подключения к TeX'у других шрифтов используется команда '`\font`'. Например, строка во входном файле

```
\font\ninerm=cmr9
```

подключит шрифт cmr9 (Computer Modern Roman 9 point). Чтобы подключить команду (например, '`\cs`') переключения шрифта, необходимо во входном файле указать

```
\font\cs=<внешнее имя шрифта>
```

или

```
\font\cs=<внешнее имя шрифта> scaled <число>
```

¹ бриллиант (3 пункта), диамант (4 пункта), перл (5 пунктов), нонпарель (6 пунктов), миньон (7 пунктов), петит (8 пунктов), боргес (9 пунктов), корпус (10 пунктов), цицero (12 пунктов), миттель (14 пунктов), терция (16 пунктов), текст (20 пунктов)

Последняя строка позволяет подключить шрифт, масштабированный в $\langle \text{число} \rangle / 1000$ раз. Обычно в наличии имеются шрифты, масштабированные в $1,2^n$ раз ($n = 0, 1, \dots, 5$). Вместо параметра $\langle \text{число} \rangle$ можно использовать $\backslash\text{magstep } n$, причем

```
\magstep0 = 1000 · 1,20 = 1000
\magstep1 = 1000 · 1,21 = 1200
\magstep2 = 1000 · 1,22 = 1440
\magstep3 = 1000 · 1,23 = 1728
\magstep4 = 1000 · 1,24 = 2074
\magstep5 = 1000 · 1,25 = 2488
```

Имеется также команда $\backslash\text{magstephalf} = 1000 \cdot \sqrt{1,2} = 1095$. Примеры:

```
\font\twelverm=xcmr10 scaled \magstep1
Шрифт xcmr101 нормального размера (\magstep0)
Шрифт xcmr10, увеличенный в 1,2 раза (\magstep1)
Шрифт xcmr10, увеличенный в 1,44 раза
```

В Т_ЕX'е имеется также возможность масштабирования всего документа (шрифты и размерные величины) сразу. Для этого в начало входного файла помещают строку вида

```
\magnification= $\langle \text{число} \rangle$ 
```

Здесь $\langle \text{число} \rangle$ имеет тот же смысл, что и выше. (Указанная команда может использоваться во входном файле только один раз.)

При подключении того или иного шрифта необходимо быть уверенным в наличии этого шрифта соответствующего увеличения.

Для подготовки какого-либо печатного издания используется комплект шрифтов, различающихся по кеглям и начертаниям, но одинаковых по рисунку. Такой комплект называется *гарнитурой* шрифта. Данный препринт набран так называемой *современной компьютерной* (computer modern) гарнитурой.

¹ в различных реализациях Т_ЕX'а во *внешних именах* шрифтов, включающих как латинские буквы, так и кириллицу, 'cm' может заменяться на 'xcm', 'lh' или оставаться без изменения

§ 4. Группы

Нередко возникает необходимость рассматривать часть входного файла как целое. В TeX'e такую часть заключают в фигурные скобки {...}. Такую конструкцию называют *группой*.

В предыдущем параграфе уже встречался пример использования группы для локального переключения шрифта. Вообще говоря, внутри группы можно не только переключать шрифт, но и определять управляющие последовательности, изменять значения параметров и т.п. Эти изменения будут локальными для данной группы, т.е. их действие прекратится с окончанием группы. В программировании группы называют «блочными структурами» или «блоками».

В TeX'e можно использовать группы для удобства работы с пробелами. Например, слово 'TeX' можно набрать как '{\TeX}' либо '\TeX{ }'. При этом не нужно беспокоиться о том, является ли следующий символ пробелом или нет. Вариант '\TeX\ ' может привести к ошибке (если следующий символ не пробел). Используя группу, можно также получить, например, три пробела подряд: '\{ }{ }{' (но не '\{ }{ }{ }').

Однако чаще группы используются для другого — для указания части текста, подвергающейся действию управляющей последовательности. Например, можно центрировать (*выключить в красную строку*) какую-либо фразу, используя команду \centerline:

```
\centerline{Это выражение центрируется.}
```

Допускается любой уровень вложенности групп. Однако нужно внимательно следить, чтобы каждой открывающей фигурной скобке соответствовала закрывающая. Вот пример вложенных групп:

```
\centerline{Это выражение {\it центрируется}.}
```

На печати получится

Это выражение *центрируется*.

Обратим внимание на то, что '\centerline' стоит снаружи фигурных скобок, а '\it' внутри. Дело в том, что \it означает смену текущего шрифта, тем самым потенциально влияя на всё последующее. \centerline действует только на ближайший символ.

Например, чтобы выключить в красную строку слово `TeX`, достаточно набрать `\centerline\TeX`, тогда как для выключки фразы ‘`TeX` имеет группы’ необходимо использовать фигурные скобки: `\centerline{\TeX\ имеет группы}`.

Другими словами наружная пара скобок в рассматриваемом примере служит для объединения нескольких слов в единый объект, а внутренняя определяет локальную блочную структуру.

§ 5. Литеры

В `TeX`’е имеется возможность получить доступ к любой букве текущего шрифта с помощью команды `\char<число>`, где *число* — числовая константа в диапазоне от 0 до 255 или управляющая последовательность, имеющая указанное значение. Например, команда `%` является сокращением для `\char37`, поскольку знак процента `%` имеет код 37.

Английская буква ‘b’ имеет код 98, поэтому слово ‘bubble’ можно набрать, например, так

```
\char98 u\char98\char98 le
```

Пробелы после ‘98’ не обязательны — они игнорируются после констант. `TeX` понимает не только десятичную форму числовых констант, но также восьмеричную и шестнадцатеричную, причем перед константой ставят соответственно перевернутый апостроф (‘) и кавычки ("). Например, восьмеричный код буквы ‘b’ — 142, а шестнадцатеричный — 62. В качестве восьмеричной константы можно использовать обратную косую черту (\) с соответствующим ASCII-кодом после нее. Так, например, ‘b’ и ‘%’ можно набрать по меньшей мере пятью способами:

```
b      \char98   \char‘142  \char"62   \char‘\b
%      \char37  \char‘45   \char"25   \char‘\%
```

Команда `\char` используется редко, поскольку почти каждому необходимому знаку уже соответствует своя управляющая последовательность (см. приложение). Иногда, правда, может потребоваться какой-нибудь экзотический знак (например, \circ \square \diamond \star \curvearrowright) тогда стоит воспользоваться командой `\char` с соответствующим кодом.

§ 6. Размеры

Т_ЕX использует *размеры*, с помощью которых можно, например, указать какой пробел нужно сделать или какой должна быть длина строки в документе. В примере параграфа 2 использовались команды ‘\vskip 1mm’ и ‘\hspace=11.3cm’, задающие соответственно пробел в 1 мм по вертикали и ширину строк 11,3 см.

В Т_ЕX’е для обозначения размеров используются следующие единицы измерения:

pt	point, пункт (интервал между строк в препринте = 12 pt)
pc	pica, пика (1 pc = 12 pt)
in	inch, дюйм (1 in = 72,27 pt)
bp	big point, большой пункт (72 bp = 1 in)
cm	centimeter, сантиметр (2,54 cm = 1 in)
mm	millimeter, миллиметр (10 mm = 1 cm)
dd	didot point (1157 dd = 1238 pt) ¹
cc	cicero, цецеро (1 cc = 12 dd)
sp	scaled point (65536 sp = 1 pt)

Физические размеры задаются в следующем виде:

$\langle \text{знак} \rangle \langle \text{число} \rangle \langle \text{единица измерения} \rangle$

или

$\langle \text{знак} \rangle \langle \text{цифры} \rangle . \langle \text{цифры} \rangle \langle \text{единица измерения} \rangle$

где $\langle \text{знак} \rangle$ — это ‘+’, ‘-’ или ничего; $\langle \text{число} \rangle$ — целое число; $\langle \text{единица измерения} \rangle$ — одна из указанных выше единиц измерения (pt, pc, in, bp, cm, mm, dd, cc, sp); $\langle \text{цифры} \rangle$ — последовательность (возможно пустая) десятичных цифр. Вместо десятичной точки ‘.’ допустимо использование десятичной запятой ‘,’.

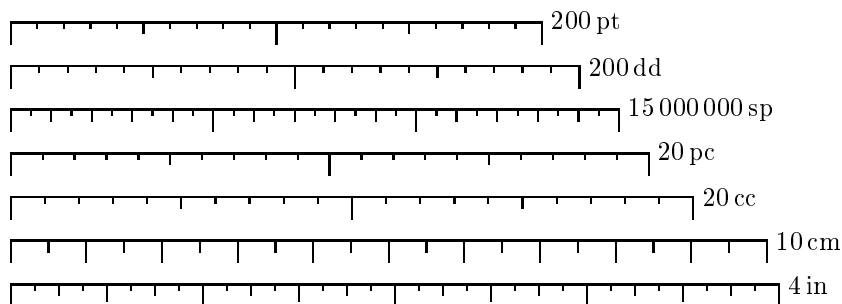
Вот несколько примеров правильно указанных размеров:

3 in	29 pc
-.013837in	+ 42,1 dd
0.mm	123456789sp

¹ 1 dd \approx 0,376 мм, что равно величине пункта, принятой у нас

Пробелы могут стоять перед знаком, числом, до и после единицы измерения — эти пробелы игнорируются. Но внутри чисел и между буквами единиц измерения пробелы ставить нельзя.

Ниже показаны «линейки», дающие наглядное представление о соизмеримости единиц измерения:



Все размеры, задаваемые указанным выше способом, относительные. Это значит, что на них влияет команда `\magnification`. Так, например, если в начале документа задано `\magnification=1200`, то все размеры (в том числе и кегль шрифтов) будут увеличены в 1,2 раза. Это очень удобно в случае, если всё нужно увеличить (или уменьшить) пропорционально¹. Однако иногда необходимо использовать «абсолютные» размеры, т.е. не подвергающиеся масштабированию. Для этого при указании размера нужно поставить ‘true’ перед единицей измерения, например,

```
\hsize = 6.5 true in
\vsizer = 24truecm
```

В принципе можно задавать любое увеличение (уменьшение) для документа. Однако при этом нужно быть уверенным в наличии соответствующих шрифтов.

В `TeX`е часто используются также такие единицы измерения, как ‘em’ (чуть больше ширины буквы ‘M’ текущего шрифта) и ‘ex’ (примерно высота буквы ‘x’ текущего шрифта). Они удобны, когда необходимо указывать размеры в масштабе кегля текущего шрифта.

¹ например этот препринт был набран с `\magnification=\magstep0` и воспроизведен в масштабе 1 : 1, но тот же самый результат получится, если указать `\magnification=\magstep2` и воспроизвести с соответствующим уменьшением

§ 7. Боксы и клей

TeX строит страницу подобно тому, как каменщик возводит стену. TeX берет «кирпичи» и надлежащим образом укладывает их, заполняя промежутки «раствором». «Кирпичи» в TeX'e называются *боксами* (box), а «раствор» — *клеем* (glue)¹.

Боксы — это прямоугольники различного размера, состоящие из отдельных литер и/или их комбинаций. Клей — это то, что заполняет на странице промежутки между боксами (*пробельный материал*).

Боксы. С боксом связываются три величины: *ширина*, *высота* и *глубина*. Ниже изображен типичный бокс с указанием так называемых *точки отсчета* (reference point) и *нижней линии* (baseline):

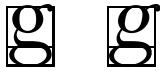


С точки зрения TeX'a каждая литера шрифта является боксом (одним из простейших видов). Каково изображение литеры, а также какими должны быть ее высота, ширина и глубина, решает разработчик шрифта. TeX же использует эти значения для размещения боксов, вычисляя в конечном итоге положение точек отсчета для всех литер на странице. Например, в шрифте cmr10 (`\rm`) буква 'h' имеет высоту 6,9444 pt, ширину 5,5555 pt и нулевую глубину, а буква 'g' — соответственно 4,3055 pt, 5 pt и 1,9444 pt.

Изображение литеры может вылезать за границы своего бокса. Это, например, намеренно делается для литер, используемых при построении больших математических знаков, таких как скобки матриц, радикалы и др. Наклонные литеры часто тоже слегка выходят за пределы своих боксов. Сравним, например, букву 'g' шрифтов cmr10 и

¹ Автор не счел возможным изменить жаргонный термин «клей», используемый в книге Д. Кнута

cmsl10 (`\rm` и `\sl`):



Для Т_ЕX’а эти буквы одинаковы, т.к. одинаковы их боксы. Его не интересует, вылезает изображение буквы за границы бокса или нет.

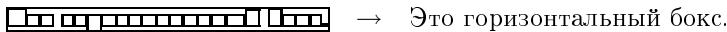
На самом деле Т_ЕX знает еще одну величину (задаваемую разработчиком шрифта) для каждой литеры — *поправку на наклон*, которая чуть больше выступающей части литеры. Например, поправка на наклон, для ‘g’ из `cmr10` равна 0,1389 pt, а из `cmsl10` — 0,8565 pt. Пробел, равный поправке на наклон, обычно вставляют с помощью ‘\’ при переходе от шрифта с наклоном к прямому шрифту (см. § 3).

Т_ЕX размещает боксы на странице двойко: в *горизонтальном* или *вертикальном* направлениях. В первом случае Т_ЕX располагает точки отсчета боксов по горизонтальной прямой, а во втором — по вертикальной. Последовательность горизонтально расположенных боксов называется *горизонтальным списком*, а вертикально — *вертикальным списком*.

Когда Т_ЕX читает абзац, то он автоматически разбивает его на строки (горизонтальные списки) и помещает их на странице в вертикальный список. Пользователь может явно задавать горизонтальные и вертикальные списки. Например, команда типа

```
\hbox{Это горизонтальный бокс.}
```

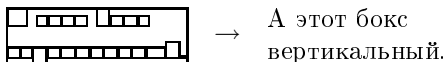
говорит Т_ЕX’у, что надо взять боксы соответствующих букв текущего шрифта и сделать из них горизонтальный бокс (h-бокс), т.е.



Из h-боксов, представляющих отдельные строки, можно сделать v-бокс (вертикальный бокс), например,

```
\vbox{\hbox{A этот бокс}\hbox{вертикальный.}}
```

Т_ЕX преобразует в



Каждая страница документа — это тоже бокс (`\vbox`), только очень большой. `TeX` создает такие боксы из вертикального списка меньших боксов, представляющих отдельные строки. В свою очередь строка — это горизонтальный список боксов отдельных литер. В более сложных случаях (математические формулы, таблицы) используются боксы внутри боксов до любого уровня вложенности. Однако пользователю редко приходится иметь дело с `h`-боксами и `v`-боксами в явном виде.

Помимо отдельных литер еще один вид простейших боксов. Это прямоугольники типа `■`, представляющие собой боксы полностью заполненные краской. Можно задавать произвольные высоту, ширину и глубину для таких боксов.

Эти прямоугольники появляются в документе обычно в виде горизонтальных или вертикальных линий. Такие линии называются *горизонтальными линейками* и *вертикальными линейками*, а в `TeX`'е им соответствуют команды `\hrule` и `\vrule`. Одинаковые черные прямоугольники можно получить, используя как `\hrule`, так и `\vrule`. Однако есть одно принципиальное различие: `\hrule` используется как элемент вертикального списка, а `\vrule` — как элемент горизонтального. Общий синтаксис указанных команд

```
\hrule height<размер> width<размер> depth<размер>
\vrule height<размер> width<размер> depth<размер>
```

Спецификации `'height <размер>'`, `'width <размер>'`, `'depth <размер>'` можно указывать в произвольном порядке или же вовсе опускать. Например, прямоугольник `■` получен с помощью `'\vrule width4pt height6pt depth2pt'`. Пример, использования `\hrule` см. в § 2. По умолчанию `\hrule` имеет высоту `0,4pt`, ширину, равную ширине текущего вертикального списка, и глубину `0pt`. Для `\vrule` высота и глубина равны высоте и глубине текущего горизонтального списка, а ширина — `0,4pt`.

Клей. Между строками текста, а также между словами имеются промежутки. Это не пустые боксы, а клей между боксами. Клей — это пробел, который обладает способностью сжиматься или растягиваться (за счет чего, например, осуществляется выравнивание строк по длине).

Клей имеет три атрибута: *пробел* (нормальная ширина), *сжимаемость* (допуск на сжатие) и *растяжимость* (допуск на растяжение)¹.

Рассмотрим пример. Пусть имеется горизонтальный список из четырех боксов (скажем, шириной 5, 6, 3 и 8 пунктов), разделенных тремя промежутками клея (для которых пробелы равны 9, 9 и 12 pt, растяжимости — 3, 6 и 0 pt, а сжимаемости — 1, 2 и 0 pt соответственно).

Суммарная ширина боксов и клея, учитывая для клея только пробелы, равна $5 + 9 + 6 + 9 + 3 + 12 + 8 = 52$ pt. Это так называемая *нормальная ширина* (natural width) горизонтального списка. Если же Т_ЭX’у дано указание сделать из этого горизонтального списка бокс шириной 58 pt, то клей должен быть растянут на 6 pt. Полная же растяжимость равна $3 + 6 + 0 = 9$ pt, поэтому Т_ЭX’ каждую из величин растяжимости умножает на $6/9$. Значит, первый промежуток клея будет $9 + (6/9) \times 3 = 11$ pt, второй — $9 + (6/9) \times 6 = 13$ pt, а третий останется неизменным.

С другой стороны, если необходимо сделать бокс шириной 49 pt, то клей должен быть сжат на 3 pt. Суммарная сжимаемость равна 3 pt, поэтому первый первый промежуток клея будет $9 - (3/3) \times 1 = 8$ pt, второй — 7 pt, а третий — 12 pt.

После того, как для всех промежутков определены их размеры, клей «затвердевает», т.е. больше не растягивается и не сжимается.

Клей может сжиматься не более, чем на величину своей сжимаемости. Значит, рассмотренный выше горизонтальный список не может быть уже 49 pt. Однако допускается растяжение клея на произвольную положительную величину.

Всё вышесказанное относится и клею в промежутках между боксами вертикального списка. Для вертикальных пробелов в Т_ЭX’е имеется несколько удобных команд. Например, получить небольшой пробел между абзацами (как между этим и следующим абзацами) нужно набрать ‘\smallskip’.

На самом деле \smallskip означает вертикальный пробел 3 pt, который может растягиваться и сжиматься на 1 pt. В Т_ЭX’е определены также ‘\medskip’ (в два раза больше, чем \smallskip) и ‘\bigskip’

¹ их английские эквиваленты — space (natural width), shrinkability и stretchability

(два раза по `\medskip`). Эти команды означают следующее:

```
\smallskip → \vskip 3pt plus 1pt minus 1pt
\medskip   → \vskip 6pt plus 2pt minus 2pt
\bigskip   → \vskip 12pt plus 4pt minus 4pt
```

Для создания вертикальных пробелов (вертикальной отбивки) в тексте рекомендуется пользоваться именно этими командами. Вообще говоря, вертикальный пробел можно задать с помощью

```
\vskip<клей>
```

где *<клей>* — это спецификация вида

```
<размер> plus<размер> minus<размер>
```

Причем ‘*plus<размер>*’ и ‘*minus<размер>*’ можно опускать (что будет означать нулевые значения); ‘*plus*’ задает величину растяжимости, а ‘*minus*’ — сжимаемости.

Аналогично задаются и горизонтальные пробелы: ‘`\hskip<клей>`’.

Интересно, что сжимаемость и растяжимость клея могут быть бесконечными. Пусть в примере с четырьмя боксами растяжимость второго промежутка клея бесконечна. Тогда суммарная растяжимость тоже бесконечна (∞). Поэтому, если сделать h-брок шириной 58 pt, то второй пробел будет $9 + (6/\infty) \times \infty = 15$ pt, а два других не изменятся.

Если такой бесконечно растяжимый клей поместить (например, командой ‘`\hfill`’) в начало горизонтального списка боксов, то они будут прижаты к правому краю, образуя h-брок; если поместить в конец, — то к левому краю; а если поместить с обоих концов, — то будут центрированы. Так, кстати, работают команды ‘`\rightline`’, ‘`\leftline`’ и ‘`\centerline`’. Если набрать

```
\rightline{Эта строка \sl выключена в правый край формата.}
\leftline{Эта строка \sl выключена в левый край формата.}
\centerline{А эта строка \sl выключена на середину формата.}
```

то на печати получится

Эта строка выключена в правый край формата.

Эта строка выключена в левый край формата.

А эта строка выключена на середину формата.

Аналогично, для вертикального списка можно использовать команду ‘\vfill’, означающую нулевой вертикальный пробел, который может бесконечно растягиваться.

В § 2 уже говорилось, что Т_ЕX вставляет дополнительный пробел в конце предложения. Но он также увеличивает растяжимость (и уменьшает сжимаемость) после знаков препинания. Наглядно это можно увидеть на следующем примере (первая строка сжата на 5 pt, вторая имеет нормальную ширину, а последующие растянуты соответственно на 5, 10, 15 и 20 pt):

Клей. Что это? Пробел, который может деформироваться.

Клей. Что это? Пробел, который может деформироваться.

Клей. Что это? Пробел, который может деформироваться.

Клей. Что это? Пробел, который может деформироваться.

Клей. Что это? Пробел, который может деформироваться.

Клей. Что это? Пробел, который может деформироваться.

Иногда в тексте встречаются сокращения типа ‘т.е.’, ‘см.’, ‘ул.’, ‘проф.’ и т.д. Точка в конце сокращения будет означать для Т_ЕX’а конец предложения. Поэтому следующий пробел будет увеличен. Чтобы этого не произошло нужно набрать ‘_’ сразу после точки. Например, ‘см. ниже’ можно набрать двояко: ‘см.~ниже’ или ‘см.\ ниже’. Оба варианта правильны, только первый запрещает Т_ЕX’у делать разрыв строки между ‘см.’ и ‘ниже’.

С другой стороны, если перед точкой стоит прописная буква, как, например, в инициалах, то Т_ЕX такую точку концом предложения не считает. Однако в конце предложения может оказаться, например, ‘НАСА.’, и последующий пробел не будет увеличен. В таком случае следует набрать ‘НАСА\hbox{.’}’ или ‘НАСА\null.’.

Если же необходимо, чтобы все пробелы после знаков препинания не отличались от обычных *междусловных* пробелов¹, то в начале входного файла следует набрать ‘\frenchspacing’. Пример:

Клей. Что это? Пробел, который может деформироваться.

Снова вернуть в режим разных пробелов можно с помощью команды ‘\nonfrenchspacing’.

¹ кстати, так обычно принято в русскоязычной литературе

§ 8. Режимы

Существует шесть *режимов* (mode) работы \TeX 'а:

- *вертикальный режим* (создание *основного вертикального списка*, из которого образуются страницы документа);
- *внутренний вертикальный режим* (создание вертикального списка внутри v-бокса);
- *горизонтальный режим* (создание горизонтального списка для абзаца);
- *ограниченный горизонтальный режим* (создание горизонтального списка внутри h-бокса);
- *математический режим* (создание математической формулы, помещаемой в горизонтальный список);
- *математический режим с выключкой* (создание математической формулы, помещаемой на отдельной строке)¹.

Для краткости будем называть последние два режима соответственно *режимом формул* и *режимом уравнений*, а под математическим режимом будем подразумевать их обобщенное название.

В простых ситуациях не обязательно знать о том, в каком из режимов работает \TeX . Но если возникает сообщение об ошибке, в котором говорится, что, например, нельзя делать того-то и того-то в ограниченном горизонтальном режиме, то в таких случаях знание режимов помогает понять и исправить ошибку.

Вернемся ко второму примеру из § 2. Вначале \TeX находится в вертикальном режиме, поэтому бокс `\hrule` и клей `\vskip 1mm` \TeX помещает в основной вертикальный список. Как только \TeX встречает букву 'O' он переходит в горизонтальный режим и делает *абзацный отступ*. Затем \TeX считывает весь абзац, делит на строки надлежащей длины, помещает строки в h-боксы, добавляет последние в основной вертикальный список и, наконец, возвращается в вертикальный режим. (Конец считываемого абзаца \TeX определил по команде `\par`.)

Далее \TeX встречает букву 'H' и снова переходит в горизонтальный режим ... Команда `\vskip` служит сигналом к окончанию абзаца

¹ соответствующие этим шести режимам английские эквиваленты: vertical mode, internal vertical mode, horizontal mode, restricted horizontal mode, math mode и display math mode

и перехода в вертикальный режим. После добавления строк абзаца в основной вертикальный список, туда же добавляется клей ‘\vskip 0.1cm’ и бокс ‘\hrule’.

Обычно в конце входного файла ставят \bye, что является сокращением для ‘\vfill\eject\end’. ‘\vfill’ переводит Т_ЕX в вертикальный режим и заполняет остаток страницы пробелом, ‘\eject’ заканчивает страницу, а ‘\end’ прекращает работу Т_ЕX’а. Файл можно заканчивать и просто командой ‘\end’.

Всякий раз, находясь в вертикальном режиме, Т_ЕX переходит в горизонтальный, когда встречает литеру, \char, \accent, \hskip, _, \vrule, \indent, \noindent¹ или признак начала математического режима (\$). Обратное в вертикальный режим Т_ЕX переключается, если встретит что-либо не совместимое с горизонтальным режимом: пустую строку, \par, \vskip и др.

Если в горизонтальном режиме Т_ЕX встречает ‘\$’, то он переходит в математический режим и обрабатывает формулу, пока не встретит закрывающий ‘\$’. Затем Т_ЕX вставляет эту формулу в текущий абзац и возвращается в горизонтальный режим. Во втором примере из § 2 знак минуса был набран в математическом режиме (\$-\$).

Если в параграфе появляется ‘\$\$’, то Т_ЕX в этом месте прерывает абзац, добавляет строки абзаца в вертикальный список, затем обрабатывает формулу в режиме уравнений, помещает формулу в h-бокс, который добавляет к вертикальному списку, а затем переходит в горизонтальный режим и обрабатывает оставшуюся часть абзаца. (Выключаемая формула должна заканчиваться на ‘\$\$’.) Пусть, например, во входном файле набрано

число $\pi = 3.1415926536\ldots$ играет важную роль.

Тогда Т_ЕX войдет в режим уравнений, обработает формулу, и в результате получится, что число

$$\pi = 3.1415926536 \dots$$

играет важную роль.

¹ \indent и \noindent обычно ставят в начале абзаца; \noindent запрещает абзацный отступ для этого абзаца, а \indent явным образом начинает абзац с отступом, равным значению параметра \parindent

Внутренний вертикальный и ограниченный горизонтальный режимы очень похожи на вертикальный и горизонтальный режимы — это построение вертикального списка внутри `\vbox`'а и горизонтального списка внутри `\hbox`'а. Есть и отличия: например, в ограниченном горизонтальном режиме Т_ЕX трактует '\$\$' как пустую формулу, а не как начало режима уравнений.

Т_ЕX помещает `\vbox` и `\hbox` в вертикальный или горизонтальный список, в зависимости от того, вертикальный или горизонтальный режим является текущим.

Будучи в вертикальном или внутреннем вертикальном режиме Т_ЕX игнорирует пробелы и пустые строки (или `\par`'ы). Однако `_` рассматривается как начало абзаца.

§ 9. Разбиение на строки и страницы

Разбиение документа на строки и страницы является одной из важнейших обязанностей наборных систем. Т_ЕX делает это автоматически, и в большинстве случаев не требуется вмешательства пользователя.

Разбиение на строки. Т_ЕX рассматривает абзац как единое целое и выбирает точки разрыва строк так, чтобы междусловные пробелы были как можно более однородными (в пределах абзаца). Причем можно управлять допустимыми размерами пробелов и переносами слов.

Обычный междусловный пробел — это клей. Его ширина зависит от текущего шрифта. По умолчанию, например, для шрифта `cmr10` (`\rm`) нормальная ширина равна 3,33 pt, сжимаемость — 1,11 pt, растяжимость — 1,67 pt. Значит минимальная ширина междусловного пробела равна 2,22 pt. Максимальная ширина зависит от значения параметра `'\tolerance'` (обозначим его t). Пробел может растягиваться на величину, в $\sqrt[3]{t/100}$ раз большую его растяжимости. Так, если $t = 100$, то ширина пробела может быть до 5 pt. В Т_ЕX'е установлено $t = 200$, поэтому пробел может достигать $3,33 + 1,67 \sqrt[3]{200/100} \approx 5,43$ pt.

Когда Т_ЕX делает h-бнок или v-бнок, то он вычисляет для них величину `'badness'` (обозначим ее b), которая показывает насколько сильно сжат или растянут клей в боксе. При разбиении абзаца на строки Т_ЕX старается сделать пробелы такими, чтобы для каждой строки удо-

влетворялось неравенство $b \leq t$. Если это оказывается невозможным, то TEX сообщает об этом пользователю (см. § 16).

В § 2 уже говорилось, что можно запретить TEX 'у делать разрыв строки между некоторыми словами. Для этого нужно вместо пробела ‘`␣`’ поставить ‘`~`’. TEX никогда не разрывает строку внутри `\hbox`'а, поэтому выражения вида ‘уравнение П–2’ следует набирать как ‘уравнение `~\hbox{П--2}`’. Можно также использовать `\hbox`, чтобы запретить перенос в каком-либо слове.

Иногда возникает необходимость заставить TEX сделать разрыв строки в каком-нибудь месте абзаца, тогда нужно набрать ‘`\break`’. Но это может привести к очень большим междусловным пробелам в строке. Если же нужно заполнить остаток строки пробелом, то вместо ‘`\break`’ нужно набрать ‘`\hfill\break`’. С другой стороны, если необходимо явно указать TEX 'у места возможных переносов в том или ином слове, то нужно использовать команду ‘`\-`’, например, ‘`кри\с\–талл`’.

Разбиение на страницы. TEX сам пытается выбрать места деления документа на страницы, и это обычно довольно хорошо получается. Однако проблема разбиения на страницы гораздо сложнее проблемы разбиения на строки, поскольку страницы обладают меньшей «эластичностью» нежели строки. Если вертикальный клей на странице имеет малые (или нулевые) растяжимость и сжимаемость, то TEX вряд ли сможет найти хорошее место начала новой страницы. Если же растяжимость и сжимаемость клея слишком велики, то страницы будут выглядеть очень неоднородными. В таких случаях необходимо вмешательство пользователя.

В этом смысле документы, в которых часто используются выключенные формулы, имеют преимущество, так как клей сверху и снизу таких формул может заметно сжиматься и растягиваться. Использование `\smallskip`, `\medskip` и `\bigskip` тоже дает TEX 'у пространство для маневра. За счет эластичности клея TEX может перенести одну или даже несколько строк с одной страницы на другую.

Иногда все-таки бывает нужно сделать принудительный обрыв страницы. Это достигается командой ‘`\eject`’. Правда, нужно опасаться появления больших вертикальных пробелов. Чтобы они не возникали, можно остаток страницы заполнить вертикальным пробелом

с помощью ‘`\vfill\eject`’ вместо ‘`\eject`’.

Нередко в документ требуется вставить громоздкие неразрывные блоки материала: рисунки, таблицы и др. Поскольку они могут занимать довольно большое место, то скорее всего возникнет проблема с их автоматическим разбиением на страницы. В Т_ЕX’е эта проблема решается с помощью механизма *вставок*. Вставка — это «плавающий» материал основного вертикального списка. Это означает, что, если вставка не умещается на текущей странице, то переносится на следующую.

В качестве вставки используется одна из конструкций

```
\topinsert<материал вставки>\endinsert
\midinsert<материал вставки>\endinsert
\pageinsert<материал вставки>\endinsert
```

В первом случае Т_ЕX попытается вставить *⟨материал вставки⟩* в начало текущей страницы, а если не хватит места, то перенесет на начало следующей страницы. Отличие второго случая от первого в том, что Т_ЕX попытается вставить *⟨материал вставки⟩* в том месте, где указано ‘`\midinsert...`’. В последнем случае *⟨материал вставки⟩* появится на отдельной странице вслед за текущей. Вставляемым материалом может быть текст, формулы, таблицы, рисунки, пропуск для резервирования места (например, ‘`\vskip 6cm`’) и т.п. (См. также § 10.)

В Т_ЕX’е имеется возможность делать сноски. Они являются специальным видом вставок. Сноску* можно получить с помощью ‘... `Сноску\footnote*{такую, как эта} можно ...`’. А в общем виде это делается так: ‘`\footnote{⟨метка сноски⟩}{⟨текст сноски⟩}`’.

§ 10. Вставка рисунков

Пусть необходимо зарезервировать место для рисунка, который будет вклеен после распечатки документа. Если набрать, например,

```
\topinsert
\hrule \vskip 4cm
\noindent{\bf Рис.\ 1.} Это подрисовочная подпись к
рис.~1, иллюстрирующему использование вставки.
Для рисунка оставлено 4~см \vskip 1mm \hrule \endinsert
```

* такую, как эта

Рис. 1. Это подрисуночная подпись к рис. 1, иллюстрирующему использование вставки. Для рисунка оставлено 4 см

то в результате вверху текущей (или следующей) страницы будет оставлено место в 4 см для рисунка. Горизонтальные линии (`\hrule`) указаны только для того, чтобы показать верхнюю и нижнюю границы вставки.

Имеется возможность вставлять рисунки непосредственно в документ, если есть соответствующие графические файлы. Для этого используется команда

```
\special{<инструкция> <имя графического файла>}
```

Формат *<инструкции>* зависит от того, какой программой выводится на экран или печатается образуемый DVI-файл (см. § 1). Например, в России и Европе очень распространен пакет `emTeX` — версия `TeX`'а для персональных компьютеров, разработанная Эберхардом Маттесом (Eberhard Mattes) из Германии. Для входящих в пакет программ просмотра и печати DVI-файлов (`dviscr.exe`, `dvihplj.exe`, `dvidot.exe` и др.) *<инструкцией>* является `'em:graph'`. Графические форматы, которые «понимает» `emTeX` — `MSP`, `PCX`, `BMF`.

Допустим у нас имеется графический файл `k1(gam).psx`, размеры которого $7,1 \times 3,7$ см (размеры можно выяснить с помощью какой-нибудь программы графического редактирования или измерить после распечатки рисунка). Чтобы поместить рисунок вверху страницы посередине, наберем следующее:

```
\topinsert  
\centerline{\vbox to 3.7truecm{\hbox to 7.1truecm
```

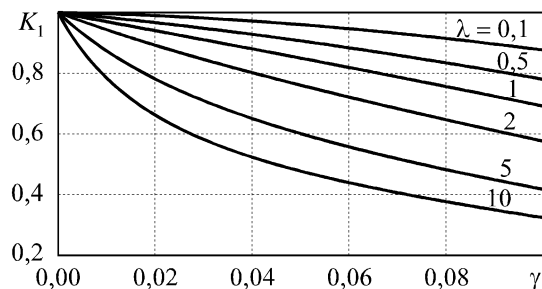


Рис. 2. Зависимость функции K_1 от γ при различных λ

```
{\special{em:graph k1(gam).psx}\hfill}\vfill}}
\medskip
\centerline{Рис.~2. Зависимость функции  $K_1$ 
от  $\gamma$  при различных  $\lambda$ }
\endinsert
```

Следует подчеркнуть, что команда ‘\special’ не умеет вычислять размеры рисунка. Она просто считает, что рисунок (точнее аргумент команды) имеет нулевую ширину и высоту, а значит, левый верхний угол изображения попадет в ту точку страницы, где стоит \special. Именно поэтому приходится использовать \vbox и \hbox с указанием соответствующих размеров, чтобы рисунок не налезал на остальной текст (‘\vbox to 3.7truecm’ создает \vbox высотой 3,7 см, а ‘\hbox to 7.1truecm’ создает \hbox шириной 7,1 см).

§ 11. Математические формулы

TeX специально предназначен для того, чтобы легко можно было вводить сложные математические выражения. Идея состоит в том, что сложные формулы разлагаются на более простые, легко соединяемые вместе, те, в свою очередь, разлагаются на еще более простые и т.д.

Простейшие формулы — это одна буква или цифра, например, ‘ x ’ и ‘ 2 ’. Чтобы их получить, нужно набрать в тексте ‘ x ’ и ‘ 2 ’. Все формулы должны заключаться в специальные математические скобки (\dots), внутри которых всё обрабатывается в математическом режи-

ме. При этом согласно обычным правилам буквы обозначают курсивные буквы (используется математический курсив, несколько отличный от обычного текстового курсива), цифры и знаки препинания — прямые цифры и знаки препинания, а дефис ‘-’ обозначает знак минуса ‘-’.

Далее в этом параграфе будет много примеров (в основном взятых из [1]), сопровождаемых при необходимости краткими пояснениями.

Все обычные пробелы в математическом режиме игнорируются:

<code>\$x\$</code>	<code>\$ x\$</code>	<code>\$ x \$</code>	<code>x</code>
<code>\$2\$</code>	<code>\$2 \$</code>	<code>\$ 2\$</code>	<code>2</code>
<code>\$(x+y)/(x-y)\$</code>	<code>\$(x + y)/(x - y)\$</code>	<code>(x + y)/(x - y)</code>	

Обратите внимание на дополнительные пробелы, окружающие знаки + и -. Т_EX об этом заботится сам.

Греческие буквы:

<code>\$\$\alpha, \beta, \gamma, \delta;\$</code>	<code>\alpha, \beta, \gamma, \delta;</code>
<code>\$\$\{\rm A}, \{\rm B}\, \Gamma, \Delta.\$</code>	<code>A, B, \Gamma, \Delta.</code>

(Наиболее употребительные команды математического режима можно найти в приложении.)

Сравните:

<code>\$\$\nu, v\$</code>	<code>\nu, v</code>
<code>\$\$\kappa, \kappa, x\$</code>	<code>\kappa, \kappa, x</code>
<code>\$\$\phi, \emptyset\$</code>	<code>\phi, \emptyset</code>
<code>\$\$\epsilon, \in\$</code>	<code>\epsilon, \in</code>

Некоторые греческие буквы имеют разный рисунок:

<code>\$\$\phi, \theta, \epsilon, \rho\$</code>	<code>\phi, \theta, \epsilon, \rho</code>
<code>\$\$\varphi, \vartheta, \varepsilon, \varrho\$</code>	<code>\varphi, \vartheta, \varepsilon, \varrho</code>

Для набора *каллиграфических* прописных букв

ABCDEFGHIJKLMN^OPQRSTUVWXYZ

используется команда переключения шрифта `\cal` (cm_sy10):

`$$\cal ABCDEFGHIJKLMNOPQRSTUVWXYZ$`

Верхний и нижний индексы обозначаются с помощью '^' и '_':

x^2	x^2	x^2y^2	x^2y^2
x_2	x_2	x_2y_2	x_2y_2
2^x	2^x	${}_2F_3$	${}_2F_3$
x^{2y}	x^{2y}	y_{x_2}	y_{x_2}
2^{2^x}	2^{2^x}	y_{x^2}	y_{x^2}
$2^{2^{2^x}}$	$2^{2^{2^x}}$	y_{m+n}	y_{m+n}

(Фигурные скобки используются для указания подформулы.)

Сравните:

$$((x^2)^3)^4 \quad ((x^2)^3)^4$$

$$\{(x^2)^3\}^4 \quad ((x^2)^3)^4$$

Допускается одновременное использование верхних и нижних индексов:

$$x_3^2 \quad x_3^2 \quad x_{92}^{31416} + \pi \quad x_{92}^{31416} + \pi$$

$$x_3^2 \quad x_3^2 \quad x_{z_c^d}^{y_a^b} \quad x_{z_c^d}^{y_a^b}$$

Для некоторых букв нижние индексы смещаются относительно нижних: '\$P_2^2\$' дает '\$P_2^2\$', но '\$P_{2^2}\$' дает '\$P_2^2\$'.

Переменные со штрихами:

$$y_1' \text{ или } y_1' \quad y_1'$$

$$y_2'' \text{ или } y_2'' \quad y_2''$$

$$y_3''' \text{ или } y_3''' \quad y_3'''$$

$$y_4^{IV} \quad y_4^{IV}$$

Радикалы (корни):

$$\sqrt{2} \quad \sqrt{2}$$

$$\sqrt{x+2} \quad \sqrt{x+2}$$

$$\sqrt{x^3 + \sqrt{\xi}} \quad \sqrt{x^3 + \sqrt{\xi}}$$

$$\sqrt[3]{2} \quad \sqrt[3]{2}$$

$$\sqrt[n]{x^n + y^n} \quad \sqrt[n]{x^n + y^n}$$

$$\sqrt[n+1]{a} \quad \sqrt[n+1]{a}$$

Математические акценты:

$$\dot{a}, \ddot{b}, \vec{v}, \tilde{x}, \hat{h} \quad \dot{a}, \ddot{b}, \vec{v}, \tilde{x}, \hat{h}$$

(Полный список приведен в приложении.)

Все символы в математическом режиме Т_ЕX делит на несколько классов и трактует их по разному.

К классу *обычных символов* Т_ЕX относит 52 буквы латинского алфавита (A...Z и a...z), означающие *переменные* (A...Z и a...z), а также 18 символов

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

К классу *бинарных операций (знаки действий)* относятся такие символы, как +, -, * и некоторые другие (см. приложение). Когда они стоят между обычными символами, Т_ЕX вставляет дополнительные пробелы:

$\$ x + y - z \$$ $x + y - z$
 $\$ x + y * z \$$ $x + y * z$
 $\$ x * y / z \$$ $x * y / z$

К *соотношениям (знаки соотношения)* относятся =, <, > и : (см. также приложение). Они также окружаются пробелами, но несколько иными (см. ниже), чем бинарные операции:

$\$x=y>z\$$ $x = y > z$
 $\$x:=y\$$ $x := y$
 $\$x\le y\ne z\$$ $x \le y \neq z$
 $\$x\sim y\sim z\$$ $x \sim y \simeq z$
 $\$x\equiv y\not\equiv z\$$ $x \equiv y \not\equiv z$
 $\$x\subset y\subseteq z\$$ $x \subset y \subseteq z$

Два символа ‘,’ (запятая) и ‘;’ (точка с запятой) трактуются Т_ЕX’ом как *знаки препинания*; после них добавляется небольшой пробел:

$\$f(x,y;z)\$$ $f(x,y;z)$

Точка считается обычным символом. Чтобы использовать двоеточие как знак препинания нужно набрать ‘\colon’:

$\$f:A\to B\$$ $f : A \rightarrow B$
 $\$f\colon A\to B\$$ $f : A \rightarrow B$

Чтобы использовать запятую как обычный символ, ее нужно взять в фигурные скобки — \TeX трактует всё, что в фигурных скобках, как обычный символ:

$$\begin{aligned} \$3,14159 r^2\$ & \quad 3,14159r^2 \\ \$3\{, \}14159 r^2\$ & \quad 3,14159r^2 \end{aligned}$$

Символы ‘(’ и ‘[’ относятся к *открывающим скобкам*, а ‘)’ и ‘]’ — к *закрывающим скобкам* (*ограничители*). Символам

$\backslash \$ \% \# \& \sim \{ \} _ \wedge$

не соответствуют никакие знаки в математическом режиме. Правда, команды ‘ $\backslash\{$ ’ и ‘ $\backslash\}$ ’ используются как скобки ‘{’ и ‘}’.

Все приведенные выше управляющие последовательности должны использоваться только в математическом режиме, иначе \TeX сообщит об ошибке (см. также раздел «Математический режим» в приложении).

В формулах часто встречаются дроби. Для их набора можно использовать команду ‘ \over ’, которая действует на всю формулу или подформулу, ограниченную фигурными скобками:

$$\begin{aligned} \$\$x+y^2\over k+1\$\$ & \quad \frac{x+y^2}{k+1} \\ \$\{x+y^2\over k\}+1\$\$ & \quad \frac{x+y^2}{k} + 1 \\ \$\$x+\{y^2\over k\}+1\$\$ & \quad x + \frac{y^2}{k} + 1 \\ \$\$x+\{y^2\over k+1\}\$\$ & \quad x + \frac{y^2}{k+1} \\ \$\$x+y^{\{2\over k+1\}}\$\$ & \quad x + y^{\frac{2}{k+1}} \end{aligned}$$

Иногда в дробях знаки уменьшаются, как будто бы они стали индексами, например,

$$\$$$1+\{x\over y\}\over 1-\{x\over y}\$\$ \quad \frac{1 + \frac{x}{y}}{1 - \frac{x}{y}}$$

Это происходит от того, что \TeX трактует формулы по-разному в зависимости от текущего *стиля*. Существует восемь стилей:

- *стиль уравнений* (для формул, выключаемых в красную строку);
- *стиль текста* (для формул, набираемых в подбор с текстом);
- *стиль индексов* (для формул, используемых как верхние или нижние индексы);
- *стиль вторых индексов* (для индексов второго порядка)¹;
- еще четыре стиля, которые отличаются от указанных тем, что *показатели степени* позиционируются чуть-чуть ниже.

Кратко эти стили будем соответственно обозначать

$$D, T, S, SS \text{ и } D', T', S', SS'.$$

Для набора математических формул \TeX использует три различных кегля шрифтов: *кегель текста*, *кегель индексов* и *кегель вторых индексов* (по умолчанию это — корпус, миньон и перл соответственно).

Для формул, набираемых в подбор с текстом ($\$. . \$$), \TeX использует стиль текста (T -стиль). Для выключаемых формул ($\$. . \$\$$) — стиль уравнений (D -стиль). В зависимости от стиля подформулы могут быть различного кегля:

<i>стиль</i>	<i>кегель</i>	<i>пример</i>
D, D', T, T'	кегель текста	$f(x) = x(ax + b)$
S, S'	кегель индексов	$f(x) = x^{ax+b}$
SS, SS'	кегель вторых индексов	$f(x) = x^{(ax+b)}$

В \TeX 'е нет SSS -стиля и соответствующего ему кегля, поэтому кегль вторых индексов является минимальным.

<i>В формуле</i>	<i>стиль верхних индексов</i>	<i>стиль нижних индексов</i>
D, T	S	S'
D', T'	S'	S'
S, SS	SS	SS'
S', SS'	SS'	SS'

Например, если ' $x^{\{a_b\}}$ ' набрано в D -стиле, то ' a_b ' будет в S -стиле, а ' b ' — в SS' -стиле; в результате получится ' x^{ab} '.

В позиционировании показателей степени для стилей D и T есть небольшая разница. ' x^2 ' дает: x^2 в D -стиле, x^2 в T -стиле и x^2 в

¹ соответствующие английские эквиваленты: display style, text style, script style, scriptscript style

стилях D' и T' . Но в отношении дробей между D - и T -стилями разница существенная:

<i>В формуле</i>	<i>стиль</i>	<i>стиль</i>
$\alpha\over\beta$ <i>стиля</i>	<i>числителя</i> α	<i>знаменателя</i> β
D	T	T'
D'	T'	T'
T	S	S'
T'	S'	S'
S, SS	SS	SS'
S', SS'	SS'	SS'

То есть, если в тексте набрать ‘ $\$1\over2\$$ ’, то это даст $\frac{1}{2}$ (единица в S -стиле, а двойка в S' -стиле), но если набрать ‘ $\$\$1\over2\$\$$ ’, то получится

$$\frac{1}{2}$$

(единица в T -стиле, а двойка в T' -стиле).

Стили можно указывать и явным образом с помощью команд `\displaystyle`, `\textstyle`, `\scriptstyle` и `\scriptscriptstyle`, которые действуют до конца формулы или подформулы. Например, ‘ $\$n+\scriptstyle n+\scriptscriptstyle n.\$\$$ ’ дает выключенную формулу

$$n + n + n.$$

(В индексных стилях TEX не вставляет вокруг знака $+$ дополнительных пробелов.)

Вот очень показательный пример, иллюстрирующий концепцию стилей:

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}}$$

Эта формула была набрана следующим образом:

```

 $\$a_0+\{1\over\displaystyle a_1+$ 
 $\{\strut 1\over\displaystyle a_2+$ 

```

$$\left\{ \frac{\displaystyle a_3}{\displaystyle a_4} \right\}$$

(`\strut` — это невидимый бокс, высота которого 8,5 pt, а глубина 3,5 pt. `\strut` используется здесь, чтобы сделать высоту числителя побольше.) Без `\strut` и `\displaystyle` получилось бы

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \frac{1}{a_4}}}}$$

Наряду с `\over` имеются также команды ‘`\atop`’ и ‘`\choose`’:

$$\text{\$}\text{\$}x\text{\@top} y+2\text{\$}\text{\$}$$

$$\frac{x}{y+2}$$

$$\text{\$}\text{\$}n\text{\choose} {1\over 2}k\text{\$}\text{\$}$$

$$\binom{n}{\frac{1}{2}k}$$

Они изменяют стили так же, как и `\over`.

В Т_ЕX’е имеется класс *больших операторов*, к которому относятся знаки суммирования \sum и интегрирования \int (а также \cup , \prod , \oint ; другие приведены в приложении), которые задаются с помощью `\sum` и `\int`. Они похожи на обычные символы, но отличаются тем, что в *D*-стиле Т_ЕX выбирает бóльший большой оператор, чем в *T*-стиле:

$$\text{\$}\text{\sum} x_n\text{\$}$$

$$\sum x_n \quad (T\text{-стиль})$$

$$\text{\$}\text{\sum} x_n\text{\$}\text{\$}$$

$$\sum x_n \quad (D\text{-стиль}).$$

Очень легко задавать пределы суммирования:

$$\text{\$}\text{\sum}_{n=1}^m\text{\$}\text{\$}$$

или

$$\text{\$}\text{\sum}^m_{n=1}\text{\$}\text{\$}$$

$$\sum_{n=1}^m$$

Однако ‘`\sum_{n=1}^m`’ дает ‘ $\sum_{n=1}^m$ ’.

В отличие от знака суммы пределы интегрирования печатаются как верхний и нижний индексы:

$$\text{\$}\text{\int}_{-\infty}^{+\infty}\text{\$}$$

$$\int_{-\infty}^{+\infty} \quad (T\text{-стиль})$$

$$\text{\$}\text{\int}_{-\infty}^{+\infty}\text{\$}\text{\$}$$

$$\int_{-\infty}^{+\infty} \quad (D\text{-стиль}).$$

Кроме того, можно задавать скобки еще большего размера, используя `\Bigl` и `\Bigr`, `\biggl` и `\biggr`, а также `\Biggl` и `\Biggr`:

$() \{\} [] \langle \rangle / \backslash \uparrow \uparrow \downarrow \downarrow \updownarrow \updownarrow$	(обычные)
$() \{\} [] \langle \rangle / \backslash \uparrow \uparrow \downarrow \downarrow \updownarrow \updownarrow$	<code>(\bigl, \bigr)</code>
$() \{\} [] \langle \rangle / \backslash \uparrow \uparrow \downarrow \downarrow \updownarrow \updownarrow$	<code>(\Bigl, \Bigr)</code>
$() \{\} [] \langle \rangle / \backslash \uparrow \uparrow \downarrow \downarrow \updownarrow \updownarrow$	<code>(\biggl, \biggr)</code>
$() \{\} [] \langle \rangle / \backslash \uparrow \uparrow \downarrow \downarrow \updownarrow \updownarrow$	<code>(\Biggl, \Biggr)</code>

В `TeX`'е имеется также встроенный механизм определения необходимой высоты скобок. Если набрать

`\left<левая скобка><подформула>\right<правя скобка>`

то `TeX` поставит вокруг подформулы указанные скобки такого размера, чтобы они закрывали эту подформулу. Например, в случае

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

`TeX` выбирает `\biggl` (и `\biggr`), поскольку меньшие скобки не закрывали бы полностью дробь.

Следует отметить, что `\left` и `\right` должны быть парными, так же как и фигурные скобки, причем конструкции такого вида, как `\left(\dots\{\dots\right)\dots\}`, запрещены. Это имеет смысл, поскольку прежде, чем вставлять скобки, `TeX` должен определить их высоту, исходя из размера заключенной между ними формулы. Кроме того, `\left` и `\right` играют роль символов группы, поэтому в приведенном примере `\over` не действует на '1+' в начале формулы.

Хотя `\left` и `\right` очень удобны, они не всегда дают желаемый результат. По меньшей мере существует три таких ситуации:

1) `\left` и `\right` могут давать скобки меньшие, чем желаемые. Сравните:

<code>\left \left x\right +\left y\right \right </code>	$ x + y $
<code>\bigl x + y \bigr </code>	$ x + y $

2) `\left` и `\right` могут давать скобки большие, чем желаемые. Сравните:

$$\begin{aligned} \text{\textbackslashleft(\sum_{k=1}^n A_k\text{\textbackslashright)}} & \left(\sum_{k=1}^n A_k \right) \\ \text{\textbackslashbiggl(\sum_{k=1}^n A_k\text{\textbackslashbiggl)}} & \left(\sum_{k=1}^n A_k \right) \end{aligned}$$

3) Иногда бывает нужно разбивать длинную выключенную формулу на несколько строк. При этом открывающие и закрывающие скобки на разных строках должны быть одной высоты. Однако из-за требования парности нельзя использовать `\left` на одной строке, а `\right` на другой. Тем не менее использование, скажем, `\Biggl` и `\Biggr` допускается.

Рассмотрим теперь некоторые более тонкие вопросы набора математических формул.

Знаки препинания. Если формула набирается в подбор с текстом, то знаки препинания, принадлежащие по смыслу предложению, должны быть снаружи знаков доллара, а принадлежащие формуле, — внутри. Но для выключенных формул знаки препинания должны быть между двойных знаков доллара. Например:

<i>Правильно</i>	<i>Неправильно</i>
Если $x < 0$, то $y = f(x)$.	Если $x < 0$, то $y = f(x)$.
для $x = a$, b или c .	для $x = a$, b или c .
$x = f(a, b)$	$x = f(a, b)$

Прямые буквы в формулах. Для переменных в формулах обычно используются курсивные и греческие буквы, но имена математических функций типа ‘sin’ всегда набираются прямым шрифтом. В `TeX`’е определены команды для таких функций, например, ‘`\sin`’, ‘`\exp`’ (другие см. в приложении). Эти команды кроме всего вставляют дополнительные пробелы вокруг названий функций:

<code>\sin2\psi=2\sin\psi\cos\psi</code>	$\sin 2\psi = 2 \sin \psi \cos \psi$
<code>O(n\ln n\ln\ln n)</code>	$O(n \ln n \ln \ln n)$
<code>\Pr(X>x)=\exp(-x/\mu)</code>	$\Pr(X > x) = \exp(-x/\mu)$
<code>\max_{1\le n\le m}\log_2 P_n</code>	$\max_{1 \leq n \leq m} \log_2 P_n$

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

Обратите внимание, что `\max` и `\lim` трактуются как большие операторы, т.е. для них индексы становятся пределами, как и для знака суммы. Это относится к `\det`, `\gcd`, `\inf`, `\lim`, `\liminf`, `\limsup`, `\max`, `\min`, `\Pr` и `\sup` в *D*-стиле.

Можно явно использовать прямой шрифт в математическом режиме с помощью `\rm`, например,

$$\begin{aligned} \sqrt{\text{Var}(Y)} & \quad \sqrt{\text{Var}(Y)} \\ x'^2 + x^2 = h = \text{const} & \quad x'^2 + x^2 = h = \text{const} \\ x_{\text{max}} - x_{\text{min}} & \quad x_{\text{max}} - x_{\text{min}} \end{aligned}$$

Пробелы между формулами. Поскольку обычные пробелы игнорируются в математическом режиме, то их надо указывать явно с помощью команд `'\qqquad'`, `'\quad'`, `'_'`. Сравните:

$$\begin{aligned} a_n = \frac{n}{n-1}, n \geq 2 & \quad a_n = \frac{n}{n-1}, n \geq 2 \\ a_n = \frac{n}{n-1}, \ n \geq 2 & \quad a_n = \frac{n}{n-1}, n \geq 2 \\ a_n = \frac{n}{n-1}, \quad n \geq 2 & \quad a_n = \frac{n}{n-1}, \quad n \geq 2 \\ a_n = \frac{n}{n-1}, \quad \quad n \geq 2 & \quad a_n = \frac{n}{n-1}, \quad \quad n \geq 2 \end{aligned}$$

`'\quad'` дает такой же пробел, как и `'_ _ _'`, а `'\qqquad'` эквивалентен `'\quad\quad'`.

Пробелы в формулах. Кроме `_`, `\quad`, `\qqquad` в *TeX*'е используются более мелкие пробелы:

- `\,` тонкий пробел (1/6 от `\quad`);
- `\>` средний пробел (2/9 от `\quad`);
- `\;` толстый пробел (5/18 от `\quad`);
- `\!` отрицательный тонкий пробел (–1/6 от `\quad`).

TeX автоматически использует эти пробелы в математических формулах. Так, например, тонкий пробел вставляется до и после имен стандартных функций ($\ln \sin \varphi$), средний пробел вставляется вокруг знаков бинарных операций ($x + y$), а толстый пробел — вокруг знаков соотношения ($m = n < k$).

Однако в ряде случаев такие пробелы необходимо указывать явно. Чаще всего используются `\`, и `\!` в таких ситуациях, как

<code>\int_0^\infty f(x)\,dx</code>	$\int_0^\infty f(x) dx$
<code>u\,dv+v\,du</code>	$u dv + v du$
<code>dx\,dy=r\,dr\,d\theta</code>	$dx dy = r dr d\theta$
<code>g=9\{,}81\rm\,м/с^2</code>	$g = 9,81 \text{ м/с}^2$
<code>h=6\{,}63\cdot 10^{-34}\rm\,Дж\,с</code>	$h = 6,63 \cdot 10^{-34} \text{ Дж с}$
<code>\frac{52!}{13!13!26!}</code>	$\frac{52!}{13!13!26!}$
<code>\sqrt{2}\,x</code>	$\sqrt{2} x$
<code>O\bigl(1/\sqrt{n}\bigr)</code>	$O(1/\sqrt{n})$
<code>\sin x\,x' \equiv x' \sin x</code>	$\sin x x' \equiv x' \sin x$
<code>x^2!/2</code>	$x^2/2$
<code>\Gamma_2 + \Delta^2</code>	$\Gamma_2 + \Delta^2$
<code>\int_0^x \int_0^y dF(u,v)</code>	$\int_0^x \int_0^y dF(u,v)$
<code>\iint_D dx\,dy</code>	$\iint_D dx dy$

Многоточие. Для набора многоточий в формулах TeX предлагается команды `\ldots` (...) и `\cdots` (...). Обычно `\cdots` ставится между знаками +, − и ×, а также между =, <, ⊂ и др. `\ldots` используется между запятыми или, например, между сомножителями, когда знаки умножения не ставятся:

<code>x_1+\cdots+x_n</code>	$x_1 + \cdots + x_n$
<code>x_1=\cdots=x_n=0</code>	$x_1 = \cdots = x_n = 0$
<code>f(x_1,\ldots,x_n)</code>	$f(x_1, \dots, x_n)$
<code>x_1x_2\ldots x_n</code>	$x_1x_2 \dots x_n$
<code>(1-x)(1-x^2)\ldots(1-x^n)</code>	$(1-x)(1-x^2) \dots (1-x^n)$

Однако, если `\ldots` и `\cdots` стоят в конце формулы или перед закрывающей скобкой типа ')', то желательно вставить тонкий пробел:

Докажем, что $(1-x)^{-1} = 1 + x + x^2 + \dots$.
 Коэффициенты c_0, c_1, \dots, c_n ненулевые.
 Пусть $a_n = 1/n^2$ ($n = 1, 2, \dots$).

Эти предложения были набраны следующим образом:

Докажем, что `(1-x)^{-1}=1+x+x^2+\cdots\,`

Коэффициенты c_0, c_1, \dots, c_n ненулевые.
Пусть $a_n = 1/n^2$ ($n=1, 2, \dots$).

Кстати, вместо ‘ \dots ’ в обоих случаях можно было бы набрать просто ‘ \dots ’.

Переносы в формулах. Обычно \TeX старается не разрывать на две строки формулы, набираемые в подбор с текстом. Однако в экстренных ситуациях \TeX может обрывать формулу после знаков соотношений типа $=, <, \rightarrow$ или (в самом крайнем случае) после знаков действия типа $+, -, \times$ при условии, что все эти знаки находятся на «внешнем уровне» формулы (т.е. не являются частью конструкций ‘ $\{ \dots \}$ ’ и ‘ $\dots \over \dots$ ’). Например, если набрать

$$f(x, y) = x^2 - y^2 = (x+y)(x-y)$$

внутри абзаца, то \TeX может разорвать формулу после знаков $=$ или же после $-, +, -$. Однако, если набрать

$$f(x, y) = \{x^2 - y^2\} = \{(x+y)(x-y)\}$$

то \TeX сможет разорвать формулу только после знаков $=$.

Всё это так, однако возникает такая проблема. В русскоязычной литературе принято ту часть формулы, которая переносится на следующую строку, начинать с того знака, которым заканчивается первая часть формулы, а по правилам \TeX 'а (и англоязычной литературы) этот знак не дублируется. В такой ситуации можно использовать конструкцию типа ‘ $\dots = \backslash\text{break} = \dots$ ’ (как крайнюю меру). Более аккуратно то же самое можно сделать так: ‘ $\dots \backslash\text{brk} = \dots$ ’, где макрос $\backslash\text{brk}$ определен следующим образом:

```
\def\brk#1{#1\discretionary{}
{\hbox{\mathsurround=0pt #1}}{}}
```

Улучшение (по сравнению с $\backslash\text{break}$) заключается в том, что $\backslash\text{brk}$ будет делать перенос только при необходимости.

Фигурные скобки. В ряде случаев желательно вставлять тонкие пробелы после открывающей и перед закрывающей фигурными скобками, обозначающими множество, например,

$$\begin{array}{ll} \{\mid x > 5\} & \{x \mid x > 5\} \\ \{x : x > 5\} & \{x : x > 5\} \end{array}$$

$$\mathop{\mathrm{bigl}}\!\left\{\mathop{\mathrm{bigl}}(x, f(x))\mathop{\mathrm{bigr}}\right. \\ \left.\mathop{\mathrm{bigm}}x\mathop{\mathrm{in}}D\mathop{\mathrm{bigr}}\right\} \quad \{ (x, f(x)) \mid x \in D \}$$

Открывающая фигурная скобка используется также для указания выбора из двух или более альтернатив, например,

$$|x| = \begin{cases} x, & \text{если } x \geq 0; \\ -x, & \text{если } x < 0. \end{cases}$$

В этом случае используется команда ‘`\cases`’:

$$\mathop{\mathrm{bigl}}\!\left\{x, \& \text{если } x \geq 0; \backslash\mathrm{cr} \\ -x, \& \text{если } x < 0. \backslash\mathrm{cr}\right\}$$

Следует отметить, что аргументом команды `\cases` является последовательность (из двух или более) строк альтернатив, в конце которых ставятся ‘`\cr`’. Каждая строка альтернативы делится на две части знаком ‘`&`’: левая часть — это формула, неявно заключенная в `$. . $.`, а правая — просто текст в ограниченном горизонтальном режиме, поэтому формулы в правой части необходимо явно заключать в `$. . $.`. Кроме того начала правой и левой частей выравниваются по вертикальной линии. Пробелы после `&` игнорируются.

Матрицы. Для набора матриц в `TeX`е используется команда ‘`\matrix`’. Например, чтобы получить

$$A = \begin{pmatrix} x - \lambda & 1 & 0 \\ 0 & x - \lambda & 1 \\ 0 & 0 & x - \lambda \end{pmatrix}.$$

следует набрать

$$\mathop{\mathrm{left}}\!\left(\mathop{\mathrm{matrix}}\{x-\lambda&1&0\backslash\mathrm{cr} \\ 0&x-\lambda&1\backslash\mathrm{cr} \\ 0&0&x-\lambda\backslash\mathrm{cr}}\right).$$

Здесь, как и в `\cases`, каждая строка заканчивается на `\cr`, а знак `&` ставится между элементами матрицы в строке. Чтобы получить круглые скобки вокруг матрицы, используются ‘`\left(`’ и ‘`\right)`’. Можно использовать и другие ограничители, например, ‘`[. . .]`’ или ‘`|| . . . ||`’. С другой стороны можно использовать команду `\pmatrix`, которая автоматически вставляет круглые скобки:

$$\mathop{\mathrm{pmatrix}}\{x-\lambda& . . . &x-\lambda\backslash\mathrm{cr}}.$$

Для матриц переменной размерности часто используются различного рода многоточия, например,

$$A = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{vmatrix}.$$

Эта матрица получена с помощью

```
$$A=\left\|\matrix{
a_{11}&a_{12}&\ldots&a_{1n}\cr
a_{21}&a_{22}&\ldots&a_{2n}\cr
\vdots&\vdots&\ddots&\vdots\cr
a_{m1}&a_{m2}&\dots&a_{mn}\cr}\right\|.$$
```

Здесь использованы новые команды `\vdots` (\vdots) и `\ddots` (\ddots).

Используя `\matrix` и различные ограничители можно набрать, например, вектор-столбец, определитель и др.

Приведенные в этом параграфе примеры далеко не исчерпывают многообразия возможностей Т_EX’а по набору математических формул различной сложности. В [1, 5, 6] можно найти примеры, в более полной мере раскрывающие эти возможности.

§ 12. Уравнения

Под *уравнениями* будем подразумевать выключенные математические формулы (набираемые отдельной строкой), а точнее конструкции вида ‘`$$...$$`’.

Однострочные уравнения. Поскольку уравнения нередко содержат текст, то может быть и чисто «текстовое» уравнение. Например, чтобы получить

“Текстовое” уравнение

достаточно набрать ‘`$$\hbox{‘‘Текстовое’’ уравнение}$$`’. Отметим, что текст, являющийся частью уравнения лучше всего помещать в `\hbox`.

Рассмотрим такой пример:

$$\|u_i\| = 1, \quad u_i \cdot u_j = 0 \quad \text{при } i \neq j.$$

Это уравнение набирается следующим образом:

```
$$|u_i|=1,\quad u_i\cdot u_j=0
\quad \text{при } i \neq j.$$
```

Отметим использование `\qquad` и `\quad`. Как правило части уравнения, логически сильнее связанные между собой, разделяются пробелом `\quad`, а связанные слабее — пробелом `\qquad`. Стоит заметить также, что последнюю часть уравнения в данном примере логически правильнее было бы набрать как `‘... \quad \text{при } i \neq j.’`

Очень часто формулы нумеруются, причем в `TeX`’е это легко делается с помощью конструкции `‘$(формула_1)\eqno(формула_2)$’`. При этом получится выключенная $\langle формула_1 \rangle$, помеченная номером $\langle формула_2 \rangle$, выключенным в правый край формата. Например, выражение

```
$$\sin 2\alpha=2\sin\alpha\cos\alpha.\eqno(2)$$
```

дает

$$\sin 2\alpha = 2 \sin \alpha \cos \alpha. \tag{2}$$

Можно поместить номер формулы и слева, если заменить `\eqno` на `\leqno`. Например,

```
$$\sin 2\alpha=2\sin\alpha\cos\alpha.\leqno(3)$$
```

дает

$$(3) \quad \sin 2\alpha = 2 \sin \alpha \cos \alpha.$$

Многострочные уравнения. В `TeX`’е имеется несколько команд, позволяющих набирать уравнения, состоящие из двух или более строк. Часто это несколько равенств, которые должны быть выровнены по знаку `=`, например,

$$\begin{aligned} X_1 + \dots + X_p &= m, \\ Y_1 + \dots + Y_q &= n. \end{aligned}$$

Такое двухстрочное уравнение лучше всего набирать с помощью команды `\eqalign`, используя уже встречавшиеся ранее `&` и `\cr`:

```
$$\eqalign{X_1+\cdots+X_p&=m,\cr
Y_1+\cdots+Y_q&=n.\cr}$$
```


В общем виде `\eqalign` применяется следующим образом:

```

\eqalign{<левая часть_1>&<правая часть_1>\cr
         <левая часть_2>&<правая часть_2>\cr
         \vdots
         <левая часть_n>&<правая часть_n>\cr}

```

где $n = 1, 2, \dots$. При этом начала правых частей выравниваются по вертикальной линии.

`\eqalign` дает бокс, центрированный по высоте, что позволяет легко набирать уравнения типа

$$\left\{ \begin{array}{l} \alpha = f(z) \\ \beta = f(z^2) \\ \gamma = f(z^3) \end{array} \right\} \quad \left\{ \begin{array}{l} x = \alpha^2 - \beta \\ y = 2\gamma \end{array} \right\}.$$

Нужно просто использовать `\eqalign` дважды:

```

$$\left\{\eqalign{
  \alpha&=f(z)\cr \beta&=f(z^2)\cr \gamma&=f(z^3)\cr}
\right\}\quad\left\{\eqalign{
  x&=\alpha^2-\beta\cr y&=2\gamma\cr}\right\}.$

```

С помощью `\eqalign` можно также пометить несколько формул одним номером, например,

$$\begin{aligned} P(x) &= a_0 + a_1x + a_2x^2 + \dots + a_nx^n, \\ P(-x) &= a_0 - a_1x + a_2x^2 - \dots + (-1)^n a_nx^n. \end{aligned} \tag{25}$$

Набрано было следующее:

```

$$\eqalign{
  P(x)&=a_0+a_1x+a_2x^2+\cdots+a_nx^n,\cr
  P(-x)&=a_0-a_1x+a_2x^2-\cdots+(-1)^na_nx^n.\cr}
\eqno (25)

```

Пусть необходимо набрать уравнение, строки которого могут помечаться номером и одновременно быть выровненными, например,

$$\begin{aligned} (x + y)(x - y) &= x^2 - xy + yx - y^2 = \\ &= x^2 - y^2; \end{aligned} \tag{7}$$

$$(x + y)^2 = x^2 + 2xy + y^2. \tag{8}$$

В этом случае удобно использовать команду `\eqalignno`. Она применяется так же, как и `\eqalign`, только для строки, помечаемой номером, нужно дополнительно указать ‘&⟨номер⟩’ непосредственно перед `\cr`. Последний пример набран так:

```


$$\begin{aligned} (x+y)(x-y) &= x^2 - xy + yx - y^2 = \\ &= x^2 - y^2; \end{aligned} \tag{7}$$


$$(x+y)^2 = x^2 + 2xy + y^2. \tag{8}$$


```

Если здесь вместо `\eqalignno` поставить `\leqalignno`, то получится

$$\begin{aligned} (x+y)(x-y) &= x^2 - xy + yx - y^2 = \\ &= x^2 - y^2; \\ (x+y)^2 &= x^2 + 2xy + y^2. \end{aligned}$$

Длинные формулы. Что делать, если встретится формула, которая не уместится на одной строке? Идея проста — нужно разбить формулу на две (или более) строки и использовать `\eqalign` или `\eqalignno`. Например, формула

$$\begin{aligned} \Phi &= 2(H_{12}\omega_1\omega_2 + H_{13}\omega_1\omega_3 + H_{23}\omega_2\omega_3) - \\ &\quad - (H_{22} + H_{33})\omega_1^2 - (H_{33} + H_{11})\omega_2^2 - (H_{11} + H_{22})\omega_3^2. \end{aligned}$$

набрана следующим образом:

```


$$\begin{aligned} \Phi &= 2(H_{12}\omega_1\omega_2 + H_{13}\omega_1\omega_3 + H_{23}\omega_2\omega_3) - \\ &\quad - (H_{22} + H_{33})\omega_1^2 - (H_{33} + H_{11})\omega_2^2 - \\ &\quad - (H_{11} + H_{22})\omega_3^2. \end{aligned}$$


```

Следует отметить использование `{}` (после минуса перед `\cr`) — это обеспечивает правильный пробел между круглой скобкой и минусом. Команда `\qqquad` сдвигает вторую строку вправо, что делает формулу более симпатичной.

§ 13. Макроопределения

Если в документе часто используется одно и то же выражение, то можно *определить* команду и набирать ее вместо этого выражения. Допустим часто встречается обозначение вектора ‘ (x_1, \dots, x_n) ’. Если набрать

```
\def\xvec{(x_1, \ldots, x_n)}
```

то `\xvec` будет сокращением для `'(x_1, \ldots, x_n)'`. (Команды, которые определяются с помощью `\def`, называются *макроопределениями* или просто *макросами*.) Тогда уравнение

$$\sum_{(x_1, \dots, x_n)} (f(x_1, \dots, x_n) + g(x_1, \dots, x_n))$$

можно набрать с помощью макроса `\xvec` очень просто:

```
$$\sum_{\xvec} \bigl(f\xvec+g\xvec\bigr)$$
```

Определение макросов позволяет не только сократить объем вводимой информации, но и уменьшает вероятность опечаток. Макрос стоит определять, если в документе одна и та же комбинация символов встречается 5—10 раз. Не бойтесь создавать свои макросы — часто это значительно облегчает набор.

Макросы могут иметь до девяти аргументов, обозначаемых при определении `#1...#9`. Так, если сделать определение

```
\def\row#1#2{(#1_1, \ldots, #1_{#2})}
```

то выражения

$$(Z_1, \dots, Z_n), \quad (\theta_1, \dots, \theta_\alpha), \quad (\tilde{Q}_1^k, \dots, \tilde{Q}_{i+j}^k)$$

можно будет получить с помощью

```
$$\row Zn, \quad \quad \quad \row\theta\alpha, \\ \quad \quad \quad \row{\tilde{\cal Q}}^k{i+j}$$
```

В `TeX`'е имеется несколько способов определения управляющих последовательностей различного типа. О двух из них уже говорилось: `\font` задает команду выбора шрифта, а `\def` определяет макрос. О других способах мы говорить здесь не будем — это тема отдельного разговора.

Пользователи `TeX`'а могут создавать свои библиотеки макросов или использовать уже разработанные. Например, можно иметь файл `macros.tex`, содержащий наиболее употребительные для пользователя макросы и команды выбора шрифтов. Чтобы их использовать, нужно указать

```
\input macros
```

в начале входного файла. Обычно большие (несколько сотен) коллекции макросов объединяют в *форматы*, такие как plain-формат Т_EX'a или lplain-формат L^AT_EX'a. Т_EX вводит формат с гораздо большей скоростью, чем просто файл макросов.

§ 14. Выравнивание

Одна из самых сложных операций, встречающихся при наборе документа, — это набор таблиц. Таблица состоит из строк и столбцов. Элементы таблицы в столбце обычно выравниваются по левому краю, центру или правому краю столбца. (Совсем не обязательно, чтобы элементы были разделены горизонтальными и/или вертикальными линейками.)

В Т_EX'e существует несколько способов набора таблиц, от простейших и до самых сложных. Рассмотрим только простейший из них.

Если набрать ‘\settabs n \columns’, то далее можно будет вводить строки, разделенные на n столбцов равной ширины. Строки задаются с помощью

```
\+⟨текст1⟩&⟨текст2⟩&... \cr
```

При этом ⟨текст₁⟩ будет начинаться с левого края первого столбца, ⟨текст₂⟩ — с левого края второго столбца и т.д. Каждая строка начинается с ‘\+’ и заканчивается на ‘\cr’. Пусть, например, набрано следующее:

```
\settabs 4 \columns
\+&&Текст третьего столбца\cr
\+&Текст второго столбца\cr
\+ \it Начало в первом столбце, а конец&&&в четвертом\cr
```

В результате получится:

		Текст третьего столбца	
	Текст второго столбца		
<i>Начало в первом столбце, а конец</i>			в четвертом

В этом примере ‘\settabs 4 \columns’ устанавливает позиции табуляции на строке через каждую четверть ширины страницы (для наглядности они указаны вертикальными линиями). Символ ‘&’ означает переход к следующей позиции табуляции.

Между столбцами нет жестких перегородок, поэтому текст одного столбца может вылезать в следующий столбец и даже за правый край страницы. Если последние столбцы в строке пустые, то можно их не указывать и поставить ‘\cr’. Т_ЕX игнорирует пробелы после ‘\+’ и ‘&’. Вокруг каждого элемента таблицы Т_ЕX неявно вставляет фигурные скобки, поэтому ‘\it’ действует только на первую часть последней строки примера.

Элементы таблицы по умолчанию выключаются в левый край столбца, но их можно также выключать по центру или в правый край столбца. Для этого нужно использовать команду \hfill (см. также § 7), например,

```
\settabs 3 \columns
\+Левый край&\hfill Центр\hfill&\hfill Правый край\cr
+\hfill Правый край&Левый край&\hfill Центр\hfill\cr
+\hfill Центр\hfill&\hfill Правый край&Левый край\cr
```

в результате дает

Левый край	Центр	Правый край
Правый край	Левый край	Центр
Центр	Правый край	Левый край

В предыдущих параграфах уже встречались ‘&’ и ‘\cr’ при использовании \cases, \eqalign и \eqalignno. Это не случайность, поскольку эти команды представляют собой специальный вид таблицы.

В Т_ЕX’е имеется очень мощное средство (команда \halign) для создания таблиц, но ограниченность объема препринта не позволяет автору описать это средство здесь.

§ 15. Атрибуты страницы

К атрибутам страницы относятся ее размеры, а также колонтитул, тело страницы, сноски, номер и т.п.

Для формирования страниц документа Т_ЕX строит основной вертикальный список, и когда тот станет достаточно большим, Т_ЕX выбирает наилучшее (со своей точки зрения) место разбиения страницы, а после этого возвращается к основному вертикальному, но предварительно Т_ЕX передает управление *подпрограмме вывода* (output routine). Эта подпрограмма «прикрепляет» к странице колонтитул,

сноски, номер и т.п., а затем «сбрасывает» полностью сформированную страницу в dvi-файл.

Ранее уже обсуждалось, как можно изменять некоторые атрибуты страницы. Например, ширину страницы можно задать с помощью команды `\hsize`, а высоту — с помощью `\vsize` (по умолчанию установлено `'\hsize=6.5in'` и `'\vsize=8.9in'`). Используя `\footnote`, можно задавать сноски.

С помощью `\hoffset` и `\voffset` можно сместить страницы по горизонтали и вертикали при печати. Например, команды

```
\hoffset=1cm \voffset=2.5cm
```

сместят страницы на 1 см вправо и на 2,5 см вниз по отношению к нормальному положению.

Иногда бывает нужно набирать документ без нумерации страниц. Для этого в начале входного файла нужно указать

```
\nopagenumbers
```

Более подробно о подпрограмме вывода и атрибутах страницы можно узнать в [1].

§ 16. Ошибки

При наборе документа как правило возникают ошибки. Это относится даже к опытным пользователям Т_ЕX'а. Поэтому важно суметь понять и исправить ошибку.

При компиляции входного файла Т_ЕX выдает на экран различного рода сообщения. Если ошибок нет, то сообщения обычно сводятся к именам считываемых `tex`-файлов и номерам обработанных страниц.

Ошибки бывают критические и некритические. После некритических ошибок Т_ЕX не останавливает работу, а лишь предупреждает пользователя, что что-то не совсем в порядке, например,

```
Overfull \hbox (29.04314pt too wide) in paragraph at lines 153--160
\tenrm ние). Ино-гда, прав-да, мо-жет по-тре-бо-вать-ся какой-нибудь|
Underfull \vbox (badness 2576) has occurred while \output is active
```

В первой строке говорится, что в абзаце между строками 153 и 160 входного файла произошло *переполнение* строки (Т_ЕX не смог найти хорошего места разрыва строки), и она вылезает за правый край

страницы на 29,04314pt (на печати справа от этого места можно увидеть черный прямоугольник). Во второй строке сообщения приводится часть строки, в которой произошло переполнение. Дефисы в словах означают места, в которых \TeX пытался сделать перенос.

Естественно, в чистовом варианте оригинал-макета не должно быть переполнений. Их можно устранить одним из следующих способов: 1) по возможности использовать ‘\’ в неуместившемся слове (в нашем случае можно указать ‘ка\–кой–ни\–будь’); 2) перефразировать предложение так, чтобы не было переполнений; 3) как крайнюю меру использовать `\break` для насильственного обрыва строки.

Сообщение ‘`Underfull \vbox (badness 2576) has ...`’ означает, что клей на текущей странице слишком сильно растянут. Поэтому надо будет посмотреть эту страницу на экране вьюером и при необходимости внести изменения, чтобы устранить недостаток. В крайнем случае можно сделать насильственный обрыв на предыдущей странице командой `\eject`.

Если произошла критическая ошибка, то \TeX выводит о ней сообщение и знак вопроса ‘?’ и останавливается в ожидании действий пользователя. Как показывает опыт, большинство ошибок — это описка в имени команды либо пропуск фигурной скобки или символа ‘\$’. Например, если вместо ‘`\vskip 0.5cm`’ набрано ‘`\vsskip 0.5cm`’, то на экране появится сообщение

```
! Undefined control sequence.
1.18 \vsskip
      0.5cm
?
```

Все сообщения о критических ошибках начинаются с символа ‘!’. Ниже указывается строка, в которой обнаружена ошибка (‘1.18’ означает ‘строка номер 18’), и неправильная команда. Следующая строка указывает то, что \TeX еще не обработал (‘0.5cm’). Знак ‘?’ означает, что \TeX ждет дальнейших указаний пользователя. В этом месте можно прекратить или продолжить обработку входного файла. В первом случае следует набрать ‘X’ и нажать [Enter]. Продолжить работу можно несколькими способами.

Если просто нажать [Enter], то \TeX продолжит обработку до следующей ошибки. Если ввести ‘R’, то \TeX продолжит работу без оста-

новки, выводя на экран сообщения о найденных ошибках. ‘Q’ означает то же, что и ‘R’, только сообщения о дальнейших ошибках на экран не выводятся. Можно также ввести ‘H’, тогда Т_ЕX выдаст более подробную информацию об ошибке (дополнительные сведения см. в [1]).

При отладке документа лучше всего ответить на знак вопроса нажатием ‘R’ или ‘Q’. Отметим, что помимо вывода на экран Т_ЕX записывает протокол работы в файл с именем входного файла и расширением `log`. Его можно внимательно изучить и исправить ошибки, обнаруженные при компиляции `tex`-файла.

Следует также иметь в виду, что Т_ЕX локализует почти все ошибки в пределах абзаца. Это очень удобно, однако некоторые ошибки (например, пропущенный символ ‘\$’) выявляются Т_ЕX’ом только по окончании абзаца. Следовательно ошибку следует искать выше строки, указываемой Т_ЕX’ом.

При исправлении ошибок лучше всего пользоваться редактором, который позволяет работать сразу с двумя файлами. Тогда в одно окно редактора следует загрузить `tex`-файл, а в другое — `log`-файл. По указанному в `log`-файле номеру строки можно будет легко найти и исправить ошибку в `tex`-файле.

Мы не будем описывать все возможные варианты ошибок. Можно посоветовать лишь обращать особое внимание на парность скобок, таких как ‘{...}’, ‘\$...\$’, ‘\$\$...\$\$’, ‘\left...\right’ и др. Пожалуй, самая коварная ошибка — это пропуск закрывающей фигурной скобки ‘}’. Иногда приходится просматривать много абзацев, чтобы найти место пропуска.

Допустим, что, наконец, Т_ЕX при компиляции входного файла не выдал ни одного сообщения об ошибке. Это, однако, не означает, что в документе всё правильно — нет только синтаксических ошибок, «отлавливаемых» Т_ЕX’ом. Необходимо еще сделать *вычитку* документа, т.е. найти (просмотрев `dvi`-файл вьюером на экране или распечатав его на принтере) и исправить возможные опечатки в тексте, формулах и другие ошибки. И только после этого можно сделать чистовой вариант оригинал-макета.

Приложение

► Спецсимволы:

<code>\</code>	<code>{\tt\char'\}</code>	<code>\</code>	<code> \\$\backslash\$</code>	<code>#</code>	<code>{\tt\char'\#}</code>	<code>#</code>	<code>\#</code>
<code>{</code>	<code>{\tt\char'\{}</code>	<code>{</code>	<code> \$\{\$</code>	<code>%</code>	<code>{\tt\char'\%}</code>	<code>%</code>	<code>\%</code>
<code>}</code>	<code>{\tt\char'\}</code>	<code>}</code>	<code> \$\}\$</code>	<code>^</code>	<code>{\tt\char'\^}</code>	<code>^</code>	<code>\^{}</code>
<code>\$</code>	<code>{\tt\char'\\$}</code>	<code>\$</code>	<code> \\$</code>	<code>_</code>	<code>{\tt\char'_}</code>	<code>_</code>	<code>_</code>
<code>&</code>	<code>{\tt\char'\&}</code>	<code>&</code>	<code> \&</code>	<code>~</code>	<code>{\tt\char'\~}</code>	<code>~</code>	<code>\~{}</code>

► Акценты, лигатуры и некоторые символы:

<code>è</code>	<code>\'e</code>	<code>š</code>	<code>\v s</code>	<code>“</code>	<code>‘</code>	<code>ffl</code>	<code>ffl</code>	<code>ø</code>	<code>\o</code>	<code>§</code>	<code>\S</code>
<code>é</code>	<code>\'e</code>	<code>ĵ</code>	<code>\H\j</code>	<code>”</code>	<code>’</code>	<code>æ</code>	<code>\ae</code>	<code>Ø</code>	<code>\O</code>	<code>¶</code>	<code>\P</code>
<code>ô</code>	<code>\^o</code>	<code>û</code>	<code>\t\i u</code>	<code>¿</code>	<code>?’</code>	<code>Æ</code>	<code>\AE</code>	<code>ı</code>	<code>\l</code>	<code>ℓ</code>	<code>\rlap/c</code>
<code>ü</code>	<code>\"u</code>	<code>ķ</code>	<code>\b k</code>	<code>¡</code>	<code>!’</code>	<code>œ</code>	<code>\oe</code>	<code>L</code>	<code>\L</code>	<code>ℒ</code>	<code>{\it\}\$}</code>
<code>ȳ</code>	<code>\=y</code>	<code>ç</code>	<code>\c c</code>	<code>ff</code>	<code>ff</code>	<code>Œ</code>	<code>\OE</code>	<code>ı</code>	<code>\i</code>	<code>℘</code>	<code>{\it\&}</code>
<code>ñ</code>	<code>\~n</code>	<code>h</code>	<code>\d h</code>	<code>fi</code>	<code>fi</code>	<code>å</code>	<code>\aa</code>	<code>ı</code>	<code>\j</code>	<code>©</code>	<code>\copyright</code>
<code>ř</code>	<code>\.p</code>	<code>--</code>	<code>--</code>	<code>fl</code>	<code>fl</code>	<code>Å</code>	<code>\AA</code>	<code>†</code>	<code>\dag</code>	<code>TeX</code>	<code>\TeX</code>
<code>ı</code>	<code>\u\i</code>	<code>---</code>	<code>---</code>	<code>ffi</code>	<code>ffi</code>	<code>ß</code>	<code>\ss</code>	<code>‡</code>	<code>\ddag</code>	<code>...</code>	<code>\dots</code>

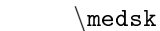
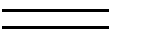
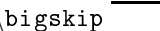
► Переключение шрифтов

<code>\rm</code>	<code>\bf</code>	<code>\it</code>	<code>\sl</code>	<code>\tt</code>
Прямой	Полужирный	Курсив	Наклонный	Равноширинный

► Горизонтальные пробелы:

На них строка может обрываться	На них строка не может обрываться
<code>\enskip</code> такой пробел	<code>\enspace</code> такой пробел
<code>\quad</code> такой пробел	<code>\thinspace</code> такой пробел
<code>\qquad</code> такой пробел	<code>\negthinspace</code> такой пробел
<code>\hskip</code> <i>⟨произвольный размер⟩</i>	<code>\kern</code> <i>⟨произвольный размер⟩</i>

► Вертикальные пробелы:

<code>\smallskip</code> 	<code>\medskip</code> 	<code>\bigskip</code> 
<code>\vskip</code> <i>⟨размер⟩</i>	<code>\kern</code> <i>⟨размер⟩</i>	<code>\vss</code> <code>\vfil</code> <code>\vfill</code>

► Разбиение на строки:

`\break` `\nobreak` `\allowbreak` `\hbox{нет переноса внутри \hbox}`
 раз\-реше\-\-ние пере\-\-носа вду\-\-три сло\-\-ва

► **Разбиение на страницы:**

`\break`, `\nobreak`, `\smallbreak`, `\medbreak`, `\bigbreak`, `\goodbreak`,
`\filbreak`, `\eject`, `\supereject`

► **Атрибуты документа:**

<code>\magnification=\magstep <i>n</i></code>	общее увеличение документа
<code>\hsize=<размер></code>	ширина текстовой страницы
<code>\vsize=<размер></code>	высота текстовой страницы
<code>\hoffset=<размер></code>	настройка левого поля страницы
<code>\voffset=<размер></code>	настройка верхнего поля страницы
<code>\baselineskip=<размер></code>	междустрочный интервал
<code>\parindent=<размер></code>	абзацный отступ
<code>\parskip=<размер></code>	межабзацный пробел
<code>\tolerance=<число></code>	допустимая растяжимость строк
<code>\headline={<текст>}</code>	верхний колонтитул
<code>\footline={<текст>}</code>	нижний колонтитул

► **Заполнение пространства по горизонтали:**

Пробелом: `\hss`, `\hfil`, `\hfill`

Линией: `\hrulefill` _____

Точками: `\dotfill`

Стрелкой влево: `\leftarrowfill` ← _____

Стрелкой вправо: `\rightarrowfill` _____ →

► **Часто используемые команды при наборе текста в нематематическом режиме:**

<code>\hsize</code> , <code>\vsize</code>	<code>\strut</code>	<code>\enspace</code>
<code>\hoffset</code> , <code>\voffset</code>	<code>\smallskip</code>	<code>\item</code> , <code>\itemitem</code>
<code>\baselineskip</code>	<code>\medskip</code> , <code>\bigskip</code>	<code>\indent</code> , <code>\noindent</code>
<code>\frenchspacing</code>	<code>\smallbreak</code>	<code>\raggedright</code>
<code>\nonfrenchspacing</code>	<code>\medbreak</code> , <code>\bigbreak</code>	<code>\raggedbottom</code>
<code>\nopagenumbers</code>	<code>\goodbreak</code>	<code>\obeylines</code>
<code>\headline</code> , <code>\footline</code>	<code>\break</code> , <code>\nobreak</code>	<code>\obeyspaces</code>
<code>\end</code> , <code>\bye</code>	<code>\allowbreak</code>	<code>\underbar</code>
<code>\hrule</code> , <code>\vrule</code>	<code>\eject</code>	<code>\hrulefill</code>
<code>\hbox</code> , <code>\vbox</code>	<code>\hfill</code> , <code>\vfill</code>	<code>\narrower</code>
<code>\line</code> , <code>\null</code>	<code>\hskip</code> , <code>\vskip</code>	<code>\rlap</code> , <code>\llap</code>
<code>\leftline</code> , <code>\rightline</code>	<code>\quad</code> , <code>\qquad</code>	<code>\adjust</code>
<code>\centerline</code>	<code>\settabs</code> , <code>\+</code> , <code>\cr</code>	<code>\if</code> , <code>\ifnum</code> , <code>\else</code> , <code>\fi</code>
<code>\rm</code> , <code>\bf</code> , <code>\it</code> , <code>\sl</code> , <code>\tt</code>	<code>\halign</code> , <code>\ialign</code>	<code>\overfullrule</code> , <code>\hfuzz</code>

Математический режим

► Строчные греческие буквы:

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

► Прописные греческие буквы:

Γ \Gamma	Δ \Delta	Θ \Theta	Λ \Lambda	Ξ \Xi	Π \Pi
Σ \Sigma	Υ \Upsilon	Φ \Phi	Ψ \Psi	Ω \Omega	
Γ {\bf\Gamma}	Δ {\bf\Delta}	...	Ω {\bf\Omega}		
Γ {\mit\Gamma}	Δ {\mit\Delta}	...	Ω {\mit\Omega}		

► Прочие символы:

\aleph \aleph	∂ \partial	\parallel \Vert	\natural \natural
\hbar \hbar	∞ \infty	\angle \angle	\sharp \sharp
\imath \imath	\prime \prime	\triangle \triangle	\clubsuit \clubsuit
\jmath \jmath	\emptyset \emptyset	\backslash \backslash	\diamondsuit \diamondsuit
ℓ \ell	∇ \nabla	\forall \forall	\heartsuit \heartsuit
\wp \wp	\surd \surd	\exists \exists	\spadesuit \spadesuit
\Re \Re	\top \top	\neg \neg, \lnot	
\Im \Im	\perp \perp	\flat \flat	

► Математические функции:

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

► Математические акценты:

\hat{a} \hat a	\check{a} \check a	\tilde{a} \tilde a	\acute{a} \acute a	\grave{a} \grave a
\dot{a} \dot a	\ddot{a} \ddot a	\breve{a} \breve a	\bar{a} \bar a	\vec{a} \vec a

► Каллиграфические буквы и «старые» цифры:

$\$cal$ ABCDEFGHIJKLM $\$$	<i>ABCDEFGHIJKLM</i>
$\$cal$ NOPQRSTUVWXYZ $\$$	<i>NOPQRSTUVWXYZ</i>
$\$1234567890$ \oldstyle1234567890 $\$$	1234567890 1234567890

► «Большие» операторы:

Σ	\sum	<code>\sum</code>	\cap	\bigcap	<code>\bigcap</code>	\odot	\bigodot	<code>\bigodot</code>
\prod	\prod	<code>\prod</code>	\cup	\bigcup	<code>\bigcup</code>	\otimes	\bigotimes	<code>\bigotimes</code>
\coprod	\coprod	<code>\coprod</code>	\sqcup	\bigsqcup	<code>\bigsqcup</code>	\oplus	\bigoplus	<code>\bigoplus</code>
\int	\int	<code>\int</code>	\vee	\bigvee	<code>\bigvee</code>	\uplus	\biguplus	<code>\biguplus</code>
\oint	\oint	<code>\oint</code>	\wedge	\bigwedge	<code>\bigwedge</code>			

► Знаки действий (помимо +, - и *):

\pm	<code>\pm</code>	\circ	<code>\circ</code>	\triangleleft	<code>\triangleleft</code>	\oplus	<code>\oplus</code>
\mp	<code>\mp</code>	\bullet	<code>\bullet</code>	\triangleright	<code>\triangleright</code>	\ominus	<code>\ominus</code>
\setminus	<code>\setminus</code>	\div	<code>\div</code>	\wr	<code>\wr</code>	\otimes	<code>\otimes</code>
\cdot	<code>\cdot</code>	\cap	<code>\cap</code>	\bigcirc	<code>\bigcirc</code>	\oslash	<code>\oslash</code>
\times	<code>\times</code>	\cup	<code>\cup</code>	\triangleup	<code>\triangleup</code>	\odot	<code>\odot</code>
$*$	<code>\ast</code>	\oplus	<code>\uplus</code>	∇	<code>\bigtriangledown</code>	\dagger	<code>\dagger</code>
\star	<code>\star</code>	\sqcap	<code>\sqcap</code>	\vee	<code>\vee, \lor</code>	\ddagger	<code>\ddagger</code>
\diamond	<code>\diamond</code>	\sqcup	<code>\sqcup</code>	\wedge	<code>\wedge, \land</code>	\amalg	<code>\amalg</code>

► Знаки соотношения (помимо <, > и =):

\leq	<code>\leq, \le</code>	\geq	<code>\geq, \ge</code>	\smile	<code>\smile</code>	\frown	<code>\frown</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\mid	<code>\mid</code>	\parallel	<code>\parallel</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\equiv	<code>\equiv</code>	\sim	<code>\sim</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	\simeq	<code>\simeq</code>	\asymp	<code>\asymp</code>
\subset	<code>\subset</code>	\supset	<code>\supset</code>	\approx	<code>\approx</code>	\cong	<code>\cong</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\bowtie	<code>\bowtie</code>	\propto	<code>\propto</code>
\sqsubseteq	<code>\sqsubseteq</code>	\sqsupseteq	<code>\sqsupseteq</code>	\models	<code>\models</code>		
\in	<code>\in</code>	\ni	<code>\ni, \owns</code>	\doteq	<code>\doteq</code>		
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\perp	<code>\perp</code>		

► Знаки соотношения с отрицанием:

$\not<$	<code>\not<</code>	$\not>$	<code>\not></code>	\neq	<code>\neq, \ne, \neq</code>
$\not\leq$	<code>\not\leq</code>	$\not\geq$	<code>\not\geq</code>	$\not\equiv$	<code>\not\equiv</code>
$\not\prec$	<code>\not\prec</code>	$\not\succ$	<code>\not\succ</code>	$\not\sim$	<code>\not\sim</code>
$\not\preceq$	<code>\not\preceq</code>	$\not\succeq$	<code>\not\succeq</code>	$\not\simeq$	<code>\not\simeq</code>
$\not\subset$	<code>\not\subset</code>	$\not\supset$	<code>\not\supset</code>	$\not\approx$	<code>\not\approx</code>
$\not\subseteq$	<code>\not\subseteq</code>	$\not\supseteq$	<code>\not\supseteq</code>	$\not\cong$	<code>\not\cong</code>
$\not\sqsubseteq$	<code>\not\sqsubseteq</code>	$\not\sqsupseteq$	<code>\not\sqsupseteq</code>	$\not\asymp$	<code>\not\asymp</code>

► Стрелки (относятся к классу соотношений):

\leftarrow	<code>\leftarrow, \gets</code>	\longleftarrow	<code>\longleftarrow</code>	\uparrow	<code>\uparrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Lleftarrow	<code>\Lleftarrow</code>	\Uparrow	<code>\Uparrow</code>
\rightarrow	<code>\rightarrow, \to</code>	\longrightarrow	<code>\longrightarrow</code>	\downarrow	<code>\downarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>	\Downarrow	<code>\Downarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>	\updownarrow	<code>\updownarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Llongleftrightarrow	<code>\Llongleftrightarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>	\nearrow	<code>\nearrow</code>
\hookrightarrow	<code>\hookrightarrow</code>	\hookleftarrow	<code>\hookleftarrow</code>	\searrow	<code>\searrow</code>
\lleftarrow	<code>\lleftarrow</code>	\rrightarrow	<code>\rrightarrow</code>	\swarrow	<code>\swarrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>	\nwarrow	<code>\nwarrow</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>		
\rightleftharpoons	<code>\rightleftharpoons</code>				

Пример использования макроса `\buildrel`:

```

 $\frac{\alpha\beta}{\text{def}}$    \buildrel \alpha\beta \over \longrightarrow
 $\text{def}$            \buildrel \rm def \over =

```

(Не путать `\over` в `\buildrel` с `\over` в дробях.)

`\iff` как и `\Longleftrightarrow` дает \iff , но с дополнительными пробелами по обеим сторонам.

► Ограничители (скобки):

<code>[</code>	<code>\lbrack,</code>	<code>[</code>	<code>{</code>	<code>\lbrace,</code>	<code>{</code>	<code><</code>	<code>\langle</code>	<code>[</code>	<code>\lfloor</code>	<code>[</code>	<code>\lceil</code>
<code>]</code>	<code>\rbrack,</code>	<code>]</code>	<code>}</code>	<code>\rbrace,</code>	<code>}</code>	<code>></code>	<code>\rangle</code>	<code>]</code>	<code>\rfloor</code>	<code>]</code>	<code>\rceil</code>
<code>/</code>	<code>/</code>	<code>\</code>	<code>\backslash</code>	<code> </code>	<code>\vert,</code>	<code>!</code>	<code>\Vert,</code>	<code>!</code>	<code>((</code>	<code>(\</code>	<code>(</code>

$((((()))$ $[[[[[]]]]]$ $\{\{\{\{\{\}}\}\}\}$ $||| ||| |||$

```

\Biggl\{\biggl\{\Bigl\{\bigl\{\{\,\}\bigl\}\Bigl\}\biggr\}\Biggr\}\quad
\Biggl[[\biggl[[\Bigl[[\bigl[[[\,]]\bigl]]\Bigl]]\biggr]]\Biggr]\quad
\Biggl\{\biggl\{\Bigl\{\bigl\{\{\,\}\bigl\}\Bigl\}\biggr\}\Biggr\}\quad
\Biggl|\biggl|\Bigl|\bigl|\{\,\}\bigl|\Bigl|\biggr|\Biggr|

```

Литература

1. Knuth D. E. The \TeX book. Addison-Wesley Publishing Company. 1984.
2. Lamport L. \LaTeX : A Document Preparation System. Addison-Wesley Publishing Company. 1986.
3. Spivak M. D. The Joy of \TeX . American Mathematical Society. 1987.
4. Samuel L. First Grade \TeX : A Beginner's \TeX Manual. Report No. STAN-CS-83-945, Stanford Department of Computer Science. 1983.
5. Бажанова Л. С. Введение в \TeX . Препринт ИФВЭ 90-116, Протвино, 1990.
6. Петрова Г. М., Руденко В. М. \TeX для начинающих. Препринт ИПМ РАН № 511, Москва, 1992.
7. Журов А. И., Карпов И. И. Основы \TeX 'а. Препринт ИПМ РАН № 518, Москва, 1992.
8. Гиленсон П. Г. Справочник технического редактора. — М.: Книга, 1978.
9. Евграфов М. А., Евграфов Л. М. \TeX : Руководство по набору и редактированию математических текстов. — М.: Физматлит, 1993.
10. Кнут Д. Е. Все про \TeX / Пер. с англ. М. В. Лисиной. — Протвино: АО $\text{RD}\text{\TeX}$, 1993.
11. Львовский С. М. Набор и верстка в пакете \LaTeX . — М.: Космосинформ, 1994.
12. Тельников К. О., Чеботарев П. З. \LaTeX . Издательская система для всех. — Новосибирск: Сибирский хронограф, 1994.

Оглавление

Введение	3
§ 1. Предварительные замечания	6
§ 2. Элементарные представления о TeX'e	7
§ 3. Шрифты	12
§ 4. Группы	15
§ 5. Литеры	17
§ 6. Размеры	17
§ 7. Боксы и клей	20
§ 8. Режимы	26
§ 9. Разбиение на строки и страницы	28
§ 10. Вставка рисунков	30
§ 11. Математические формулы	32
§ 12. Уравнения	47
§ 13. Макроопределения	50
§ 14. Выравнивание	52
§ 15. Атрибуты страницы	53
§ 16. Ошибки	54
Приложение	57
Литература	62

Алексей Иванович Журов

Основы ТрХа

Подписано к печати ???.??.??. Заказ № ???-??. Тираж ??? экз.

Отпечатано на ротапринтере Института проблем механики РАН
117526, Москва, пр-т Вернадского 101