மனோன்மணியம் சுந்தரனார் பல்கலைக்கழகம்

MANONMANIAM SUNDARANAR UNIVERSITY
TIRUNELVELI-627 012

தொலைநிலை தொடர் கல்வி இயக்ககம்

DIRECTORATE OF DISTANCE AND
CONTINUING EDUCATION

B.Sc., Mathematics (I Year)

LaTeX

Study Learning Material Prepared by
Dr. I. Valliammal, M.Sc., M.Phil., Ph.D.,
Assistant Professor of Mathematics,
Manonmaniam Sundaranar University, Tirunelveli-12

Sub. Code: JSMA22 **(For Private Circulation only)**

B.Sc., Mathematics – I YEAR

JSMA22 - LaTeX

SYLLABUS

Unit-I:

Typing text: Words, sentences and paragraphs - symbols not on the keyboard - comments and footnotes - Changing font Characteristics - Lines, paragraphs and pages - spaces - Boxes.

Unit-II:

Text environments: some general rules for displayed text environments - List of environments-style and size environments - proclamations (theorem-like structures) - Proof environments - Tabular environments - Tabbing environments - Miscellaneous displayed text environments.

Unit-III:

Typing math: Math environments - spacing rules - equations - spacing rules - equations - Basic constructs - Arithmetic operations - Delimiters – Operators - Math accents - Stretchable horizontal lines - formula gallery.

Unit-IV:

More math: Spacing of symbols building new symbols - math alphabets and symbols - vertical spacing - Tagging and grouping-Generalized Fractions - Boxed formulas.

Unit-V:

Latex documents: The structure of a document - The preamble - Abstract - Sectioning - Cross referencing - Bibliographies.

Recommended Text:

George Gratzer, More Math into LaTeX,4th edition, Springer, 2007


KNOWLEDGE IS POWER

JSMA22 - LaTeX

CONTENTS

UNIT-1

UNIT-2

UNIT-3

Unit-I:

Typing text: Words, sentences and paragraphs - symbols not on the keyboard - comments and footnotes - Changing font Characteristics - Lines, paragraphs and pages - spaces - Boxes.

1.0 Typing text:

Introduction:

In Unit 1, we briefly discussed how to type text in a document. Now we take up this topic more fully.

This unit starts with a discussion of the keyboard and continues with the rules for spaces. We cover a very important topic that must precede any in-depth discussion of LATEX, how to control LATEX with commands and environments.

A document may contain symbols that cannot be found on your keyboard. Then, we show how to get these symbols in our typeset documents by using commands.

Some other characters are defined by LATEX as command characters. For example, the % character plays a special role in the source document. Also, you will see how % is used to comment out lines. And, we introduce the command for footnotes.

Then, we discuss the commands (and environments) for changing fonts, their shapes and sizes and learn about lines, paragraphs, and pages.

The judicious use of horizontal and vertical spacing is an important part of document formatting, and also learn how to typeset text in a “box”, which behaves as if it were a single large character.

To help the discussion along, we shall use the terms *text mode* and *math mode* to distinguish between typesetting text and math.

The keyboard:

Most of the keys on your computer’s keyboard produce characters, while others are function or modifier keys.

Basic keys:

The basic keys are grouped as follows:

Letters:

The 52 letter keys:

a b c ... z A B C ... Z

Digits:

The ten digits:

1 2 ... 9 0

Old-style digits are available with the *\oldstylenums* command. The next line shows the default digits followed by the old style digits:

1234567890 $^{12}345^{6}7^{8}90$

typed as

1234567890 *\quad \oldstylenums*{1234567890}

Punctuation:

There are nine punctuation marks:

, ; . ? ! : ‘ ’ -

The first six are the usual punctuation marks. The ‘ is the *left single quote*—also known as the *grave accent*—while ’ doubles as the *right single quote* and *apostrophe*. The - key is the *dash* or *hyphen.*

Parentheses:

There are four:

( ) [ ] (and) are *parentheses;* [and] are called (*square*) *brackets.*

Math symbols:

Seven math symbols correspond to keys. The math symbols are:

∗ + = − < > /

The last four characters have a role also in text mode:

- The minus sign − corresponds to the hyphen key.
- The math symbols < and > correspond to the keys < and >; use them only in math mode.

Note that there is also a version of colon (:) for math formulas.

Space keys:

Pressing the space bar gives the *space character*. Pressing the tab key gives the *tab character*. When typesetting the source file, LATEX does not distinguish between these two. Pressing the *Return* key gives the *end-of-line character*. These keys produce *invisible characters* that are normally not displayed on your monitor by the text editor. Different computer systems have different end-of-line characters, which may cause some problems when transferring files from one system to another. A good text editor translates end-of-line characters automatically or on demand. When explaining some rules of LATEX, sometimes it is important to show if a space is required. In such cases, I use the symbol verb* ␣ to indicate a space, for instance, \in␣ut and \␣.

The tilde ˜ produces a *nonbreakable space* or *tie*.

Special keys:

There are 13 special keys on the keyboard:

$ % & ~ _ ^ \ { } @ " |

They are mostly used to give instructions to LATEX and some are used in math mode, and some in BIBTEX on how to print these characters in text. Only @ requires no special command, type @ to print @.

Prohibited keys:

Keys other than those discussed in previous are prohibited! Specifically, do not use the computer's modifier keys— Control, Alt, Escape, and others—to produce special characters, such as accented characters. LATEX will either reject or misunderstand them. Prohibited characters may not cause problems in some newer

LATEX implementations. They may just print ï£¡ if your source file has ï£¡, and ignore the invisible invalid characters. However, for portability reasons, you should avoid using prohibited characters. The babel package provides support for using some modifier keys.

If there is a prohibited character in your document, you may receive a message such as

! Text line contains an invalid character.

l.222 completely irreducible^^?

^^?

Delete and retype the offending word or line until the error goes away.

1.1 Words, sentences, and paragraphs:

Text consists of words, sentences, and paragraphs. In text, *words* are separated by one or more spaces, which may include a single end-of-line character (see the rule, **Spacing in text**), or by parentheses and punctuation marks. A group of words terminated by a period, exclamation point, or question mark forms a *sentence* (not all periods terminate a sentence). A group of sentences terminated by one or more blank lines constitutes a *paragraph.*

1.1.1 Spacing rules:

Here are the most important LATEX rules about spaces in text in the source file.

Rules of Spacing in text:

1. Two or more spaces in text are the same as one.
2. A tab or end-of-line character is the same as a space.
3. A blank line, that is, two end-of-line characters separated only by spaces and tabs, indicates the end of a paragraph. The *\par* command is equivalent.
4. Spaces at the beginning of a line are ignored.

Rules 1 and 2 make cutting and pasting text less error-prone. In our source file, we do not have to worry about the line length or the number of spaces separating words or sentences, as long as there is at least one space or end-of-line character separating any two words. Thus

We do not have to worry
about the number of spaces
separating words, as long as there
is at least one space or end-of-line character
separating any two words.

produces the same typeset text as

We do not have to worry about the number of spaces
separating words, as long as there is at least one space
or end-of-line character separating any two words.

However,

the number of spaces separating words,
as long

and

the number of spaces separating words,
as long

produce different results:

the number of spaces separating words, as long
the number of spaces separating words, as long

Notice the space between “words” and the comma in the second line. That space was produced by the end-of-line character in accordance with Rule 2.

It is very important to maintain the readability of your source file. LATEX may not care about the number of spaces or line length, but you, your coauthor, and your editor might.

Rule 3 contradicts Rules 1 and 2, consider it an exception. Sometimes—especially when defining commands and environments—it is more convenient to indicate the end of a paragraph with *par*.

1.1.2 Periods:

LATEX places a certain size space between words—the *interword space*—and a somewhat larger space between sentences—the *intersentence space.* To know which space to use, LATEX must decide whether or not a period indicates the end of a sentence.

Rule 1:

Period

To LATEX, a period after a capital letter, for instance, A. or caT., signifies an abbreviation or an initial. Generally, every other period signifies the end of a sentence. This rule works most of the time. When it fails—for instance, twice with e.g.—you need to specify the type of space you want, using the following two rules.

Rule 2:

Period

If an abbreviation does not end with a capital letter, for instance, etc., and it is not the last word in the sentence, then follow the period by an interword space (\␣) or a tie (˜), if appropriate.

Recall that \␣ provides an interword space.

The result was first published, in a first approximation,
in the Combin.\ Journal. The result was first published,

in a first approximation, in the Combin. Journal.

prints as

The result was first published, in a first approximation, in the Combin. Journal.
The result was first published, in a first approximation, in the Combin. Journal.

Notice that Combin. in the first line is followed by a regular interword space. The intersentence space following Combin. in the second line is a little wider; it is an error.

A tie (or nonbreakable space)—is more appropriate than \␣ in phrases such as Prof. Smith, typed as Prof.~Smith, and pp. 271–292, typed as pp.~271--292.

The thebibliography environment handles periods properly. You do not have to mark periods for abbreviations (in the form .\␣) in the name of a journal, so

Acta Math. Acad. Sci. Hungar.

is correct.

Rule 3:

Period

If a capital letter is followed by a period and is at the end of a sentence, precede the period with \@.

For example,

(1) follows from condition~H\@. We can proceed
(1) follows from condition~H. We can proceed

prints:

(1) follows from condition H. We can proceed
(1) follows from condition H. We can proceed

Notice that there is not enough space after H. in the second line.

Most typographers agree on the following rule (see, for instance, *The Elements of Typographic Style* by Robert Bringhurst [8], p. 30):

Rule 4:

Period

Add no space or a thin space (\,) within strings of initials and be consistent.

So W.H. Lampstone with no space or W. H. Lampstone with thin space is preferred over W. H. Lampstone. My personal choice is W. H. Lampstone with thin space.

To make all intersentence spaces equal to the interword space—as required in French typography—we can use the command

\frenchspacing

To switch back to using spaces of different sizes, give the command

\nonfrenchspacing

1.1.3 Commanding LATEX:

How do we command LATEX to do something special for us, such as starting a newline, changing emphasis, or displaying the next theorem? We use *commands* and special pairs of commands called *environments*.

Most, but not all, commands have *arguments,* which are usually fairly brief. Environments have *contents*, the text between the *\begin* and *\end* commands. The contents of an environment can be several paragraphs long.

Commands and environments:

The *\emph{text } command* instructs LATEX to emphasize its argument, *text.*

The \& command has no argument; it instructs LATEX to typeset &.

The flushright *environment* instructs LATEX to right justify the content, the text between the two commands

\begin{flushright}

\end{flushright}

The content of the document environment is the body of the article and the content of the abstract environment is the abstract.

Environments:

An environment starts with the command

\begin{name }

and ends with

\end{name }

Between these two commands is the *content* of the environment, affected by the definition of the environment.

Commands:

A LATEX command starts with a backslash, \, and is followed by the *command name.* The *name* of a command is either a *single non-alphabetic character* other than a tab or end-of-line character or a *string of letters*, that is, one or more letters.

So # and ’ are valid command names. The corresponding commands \# and \’ are input and date are also valid command names. However, input3, in#ut, and in␣ut are not valid names because 3, #, and ␣ should not occur in a multicharacter command name. Note that ␣ is a command name, the command \␣ produces a blank.

LATEX has a few commands, for instance, $ that do not follow this naming scheme, that is, they are not of the form *name*.

Command termination:

LATEX finds the end of a command name as follows:

- If the first character of the name is not a letter, the name is the first character.

- If the first character of the name is a letter, the command name is terminated by the first nonletter.

If the command name is a string of letters, and is terminated by a space, then LATEX discards all spaces following the command name.

While input3 is an invalid name, \input3 is not an incorrect command. It is the \input command followed by the character 3, which is either part of the text following the command or the argument of the command.

*LATEX also allows some command names to be modified with *. Such commands are referred to as *-ed commands. Many commands have *-ed variants. \hspace* is an often-used *-ed command.*

Command and environment names:

Command and environment names are *case sensitive. \ShowLabels* is not the same as *showlabels*.

Arguments:

Arguments are enclosed in braces, {}.

Optional arguments are enclosed in brackets, [ ].

Commands may have *arguments,* typed in braces immediately after the command. The argument(s) are used in processing the command.

Accents provide very simple examples. For instance, \'{o}—which produces ó—consists of the command \' and the argument o. In the command

\bibliography{article1}

the command is *\bibliography* and the argument is article1.

Sometimes, if the argument is a single character, the braces can be dropped: \'o also typesets as ó.

Some environments also have arguments. For example, the *alignat environment* is delimited by the commands

\begin{alignat}{2}

and

\end{alignat}

The argument, 2, is the number of columns—it could be any number 1, 2, . . . A command or environment may have more than one argument. The *\frac* command has two, *$\frac{1}{2}$* typesets as $\frac{1}{2}$. The custom command *\con* has three.

Some commands and environments have one or more *optional arguments*, that is, arguments that may or may not be present. The *\sqrt* command has an optional argument for specifying roots other than the square root. To get $\sqrt[3]{25}$, type *\sqrt[3]{25}*. The *\documentclass* command has an argument, the name of a document class, and an optional argument, a list of options, for instance, *\documentclass[12pt,draft,leqno]{amsart}*

If we get an error when using a command, check that:

1. The command is spelled correctly, including the use of uppercase and lowercase letters.
2. We have specified all required arguments in braces.
3. Any optional argument is in brackets, not braces or parentheses.
4. The command is properly terminated.
5. The package providing the command is loaded with the *\usepackage* command.

Most errors in the use of commands are caused by breaking the termination rule. We can illustrate some of these errors with the *\today* command, which produces today's date. The correct usage is

\today\ is the day

or

\today{ } is the day

which both typeset in the following form

July 19, 2015 is the day

In the first case, *today* was terminated by \␣, the command that produces an interword space. In the second case, it was terminated by the *empty group* { }.
If there is no space after the \today command, as in
todayis␣*the*␣*day*

we get the message

! Undefined control sequence.
l.3 \todayis
the day

LATEX thinks that *todayis* is the command, and, of course, does not recognize it.

If you type one or more spaces after \today:

\today␣ ␣is␣the␣day

LATEX interprets the two spaces as a single space by the first space rule, and uses that one space to delimit \today from the text that follows it. So LATEX produces

July 19, 2015is the day

If a command—or environment—can have an optional argument and

- none is given, and
- the text following the command starts with [,
  then type this as {[}.

Scope:

A command issued inside a pair of braces { } has no effect beyond the right brace, except for the rare *global* commands. We can have any number of pairs of braces:
{ ... { ... { ... } ... } ... }

The innermost pair containing a command is the *scope* of that command. The command has no effect outside its scope. We can illustrate this concept using the *\bfseries* command that switches the font to boldface:

{some text *\bfseries bold text*} no more bold

typesets as

some text **bold text** no more bold

The commands *\begin{name }* and *\end{name }* bracketing an environment act also as a pair of braces. In particular, $, \[, and \] are special braces.

Rules of Braces:

1. Braces must be balanced: An opening brace has to be closed, and a closing brace must have a matching opening brace.

2. Pairs of braces cannot overlap.

If we have one unmatched closing brace, we get a message such as

! Too many }'s

If special braces, say, *\begin{name}* and *\end{name}*, do not balance, we get an error message:

! LaTeX Error: \begin{name} on input line 21

ended by \end{document}.

or

! LaTeX Error: \begin{document} ended by \end{name}.

To illustrate the second rule, here are two simple examples of overlapping braces.

Example 1:

{\bfseries some text

\begin{lemma}

more text} final text

\end{lemma}

Example 2:

{some \bfseries text, then math: $\sqrt{2} }, \sqrt{3}$

In Example 1, the scope of *\bfseries* overlaps the braces *\begin{lemma}*, *\end{lemma}*.

In Example 2, the scope of *\bfseries* overlaps the special braces $ and $.

Example 1 is easy to correct:

{\bfseries some text}

\begin{lemma}

{\bfseries more text}

final text

\end{lemma}

Example 2 may be corrected as follows:

{some \bfseries text, then math:} $\sqrt{2}, \sqrt{3}$

Actually, √2 does not even have a bold version.

If the braces do overlap and they are of the same kind, LATEX simply misunderstands the instructions. The closing brace of the first pair is regarded as the closing brace of the second pair, an error that may be difficult to detect. LATEX can help if special braces overlap. Typesetting Example 1 gives the message

! Extra }, or forgotten \endgroup.

l.7 more text }

final text

Types of commands:

It may be useful at this point to note that commands can be of various types. Some commands have arguments, and some do not. Some commands effect change only in their arguments, while some commands declare a change.

For instance, *\textbf{This is bold}* typesets the phrase This is bold in bold type: **This is bold** and has no effect on the text following the argument of the command. On the other hand, the command *\bfseries* declares that the text that follows should be bold. This command has no argument. I call a command that declares change a *command declaration.* So *\bfseries* is a command declaration, while *\textbf* is not. As a rule, command declarations are commands without arguments.

Commands with arguments are called *long* if their argument(s) can contain a blank line or a *\par* command; otherwise, they are *short.* For example, *\textbf* is a short command. The *\parbox* command is long.

Finally, the effect of a command remains within its scope. This is true only of *local* commands. There are also some *global* commands, such as the *\setcounter* command.

Fragile commands:

As a rule, LATEX reads a paragraph of the source file, typesets it, and then goes on to the next paragraph. Some information from the source file, however, is separately stored for later use. Examples include the title of an article, which is reused as a running head; titles of parts, sections, subsections, and other sectioning commands, which are used in the table of contents; footnotes; table and figure captions, which are used in lists of tables and figures; and index entries.

These are *movable arguments*, and certain commands embedded in them must be protected from damage while being moved. LATEX commands that need such protection are called *fragile.* The inline math delimiter commands \(and \) are fragile, while $ is not.

In a movable argument, fragile commands must be protected with a *\protect*

command. Thus

The function \(f(x^{2}) \)

is not an appropriate section title, but

The function \protect \(f(x^{2}) \protect \)

is. So is

The function $f(x^{2})$

To be on the safe side, we should protect every command that might cause problems in a movable argument. Alternatively, use commands declared with *DeclareRobustCommand*

This command works the same way as *newcommand* but the command defined is *robust*, that is, not fragile.

1.2 Symbols not on the keyboard:

A typeset document may contain symbols that cannot be typed. Some of these symbols may even be available on the keyboard but you are prohibited from using them. In this section, we discuss the commands that typeset some of these symbols in text.

1.2.1 Quotation marks:

To produce single and double quotes, as in

‘subdirectly irreducible’ and *“subdirectly irreducible”*

type

‘subdirectly irreducible’ and *‘‘subdirectly irreducible’’*

Here, ‘ is the left single quote and ’ is the right single quote. Note that the double quote is obtained by pressing the single quote key twice, and *not* by using the double quote key. If we need single and double quotes together, as in “She replied, ‘No.’ ”, separate them with \, (which provides a thin horizontal space):

‘‘She replied, ‘No.’\,’’

1.2.2 Dashes:

Dashes come in three lengths. The shortest dash, called a *hyphen*, is used to connect words:

Mean-Value Theorem

This phrase is typed with a single dash:

Mean-Value Theorem

A medium-sized dash, called an *en dash*, is typed as -- and is used

- For number ranges; for instance, the phrase see pages 23–45, is typed as see pages˜23--45

Note: ˜ is a nonbreakable space or tie.

- In place of a hyphen in a compound adjective when one of the elements of the adjective is an open compound (such as New York) or hyphenated (such as non-English).

For instance, the phrase Schmidt–Freid adjoint, is typed as

Schmidt--Freid adjoint

A long dash—called an *em dash*—is used to mark a change in thought or to add emphasis to a parenthetical clause, as in this sentence. The two em dashes in the last sentence are typed as follows:

A long dash---called an *emph{em dash}*---is used

In math mode, a single dash is typeset as theminus sign − (a binary operation) with some spacing on both sides, as in 15 − 3 or the “negative” as in −3.

Note that there is no space before or after an en dash or em dash.

1.2.3 Ties or nonbreakable spaces:

A *tie* or *nonbreakable space* or *blue space* is an interword space that cannot be broken across lines. For instance, when referencing P. Neukomm in an article, we do not want the initial P. at the end of a line and the surname Neukomm at the beginning of the next line. To prevent such an occurrence, we should type P.~Neukomm.

If our keyboard does not have ~, use the \nobreakspace command instead, and type P.*nobreakspace* Neukomm. The following examples show some typical uses:

Theorem~\ref{T:main} in Section~\ref{S:intro}

Donald~E. Knuth

assume that $f(x)$ is (a)~continuous, (b)~bounded

the lattice~L

Sections~\ref{S:modular} and~\ref{S:distributive}

In~L, we find
Of course, if we add too many ties, as in
Peter~G.~Neukomm% Incorrect!
LATEX may send you a line too wide message.

The tie (~) absorbs spaces, so typing P. ␣ ~ ␣ Neukomm works just as well. This feature is convenient when you add a tie during editing.

1.2.4 Special characters:

The characters corresponding to nine of the 13 special keys are produced by typing a backslash (\) and then the key, as shown in Table 1.1.

If for some reason we want to typeset a backslash in your document, type the command *textbackslash*, which typesets as \. We might think that we could get a typewriter style backslash by utilizing the *texttt* command

\texttt{\textbackslash}

but this is not the case, *\textbackslash* and *\texttt{\textbackslash}* produce the same symbol, \, which is different from the typewriter style backslash: \. Look at them side-by-side: \ \. For a typewriter style backslash, you can use the *\bsl* command or the *\texttt{\symbol{92}}* command introduced later.

The | key is seldom used in text. If we need to typeset the math symbol | in text, type *\textbar*.

Name	Type	Typeset
Ampersand	\&	&
Caret	\^{}	^
Dollar Sign	\$	$
Left Brace	\{	{
Right Brace	\}	}
Underscore (or Lowline)	_	_
Octothorp	\#	#
Percent	\%	%
Tilde	\~{}	~

Table 1.1: Nine special characters.

Note that in text, * typesets as *, whereas in a formula it typesets centered as $*$. To typeset a centered star in text, use the command *\textasteriskcentered*. And @ typesets as @.

Finally, the " key should never be used in text. See Section 1.4.1 for the proper way to typeset double quotes. Nevertheless, sometimes " may be used to typeset ", as in the computer code segment print("Hello!"). In BIBTEX and *MakeIndex*, " has special meanings.

Be careful when typing \{ and \} to typeset the braces { }. Typing a brace without its backslash results in unbalanced braces, in violation of the first brace rule.

We can also produce special characters with the *\symbol* command:

\symbol{94} typesets as ^

\symbol{126} typesets as ˜

\texttt{\symbol{92}} typesets as \

The argument of the *\symbol* command is a number matching the slot of the symbol in the layout (encoding) of the font. The layout for the Computer Modern typewriter style font is shown in Table 1.2.

Alternatively, instead of *\texttt{\symbol{92}}*, we can use *\texttt{\char'\\}*

	0	1	2	3	4	5	6	7	8	9
x	`	´	ˆ	˜	¨	″	°	ˇ	˘	¯
1x	·	¸	˛	¸	‹	›	“	”	„	«
2x	»	–	—		_	ı	j	ff	fi	fl
3x	ffi	ffl	␣	!	"	#	$	%	&	’
4x	(	)	*	+	,	-	.	/	0	1
5x	2	3	4	5	6	7	7	9	:	;
6x	<	=	>	?	@	A	B	C	D	E
7x	F	G	H	I	J	K	L	M	N	O
8x	P	Q	R	S	T	U	V	W	X	Y
9x	Z	[	\	]	^	_	‘	a	b	c
10x	d	e	f	g	h	i	j	k	l	m
11x	n	o	p	q	r	s	t	u	v	w
12x	x	y	z	{	|	}	~	-		

Table 1.2: Font table for the Computer Modern typewriter style font.

Any character x in the font can be accessed by typing the character itself as *‘\x*. This way we don’t have to look up the position of the symbol.

We can obtain similar tables for any font in your LATEX implementation by using the *fonttbl.tex* file in your samples folder.

For more about font tables, see the *nfssfont.tex* file, part of the standard LATEX distribution.

1.2.5 Ellipses:

The text ellipsis, . . . , is produced using the *\dots* command. Typing three periods produces ... (notice that the spacing is wrong). *\dots* is one of several commands that can be used to create ellipses in formulas.

1.2.6 Ligatures:

Certain groups of characters, when typeset, are joined together—such compound characters are called *ligatures.* There are five ligatures that LATEX typesets automatically (if we use the Computer Modern fonts): ff, fi, fl, ffi, and ffl. If we want to prevent LATEX from forming a ligature, separate the characters with the command *\textcompwordmark*. Compare *iff* with *if f*, typed as *iff* and

if\textcompwordmark f

Enclosing the second character in braces ({}) is a crude method of preventing the ligature.

1.2.7 Accents and symbols in text:

LATEX provides 15 European accents. Type the command for the accent (\ and a character), followed by the letter (in braces) on which we want the accent placed (see Table 1.3). For example, to get Grätzer György, type

Gr\"{a}tzer Gy\"{o}rgy

and to get Ö type *\"{O}.*

To place an accent on top of an i or a j, you must use the *dotless* version of i and j. These are obtained by the commands \i and \j: \’{\i} typesets as í and \v{\j} typesets

as ˇj. Tables 1.4 and 1.5 list some additional text symbols and European characters available in LATEX when typing text.

Note that the *textcircled* command (in Table 1.5) takes an argument.

Name	Type	Typeset	Name	Type	Typeset
acute	\'{o}	ó	macron	\={o}	ō
breve	\u{o}	ŏ	overdot	\.{g}	ġ
caron/haček	\v{o}	ǒ	ring	\r{u}	ů
cedilla	\c{c}	ç	tie	\t{oo}	o͡o
circumflex	\^{o}	ô	tilde	\~{n}	ñ
dieresis/umlaut	\"{u}	ü	underdot	\d{m}	ṃ
double acute	\H{o}	ő	underbar	\b{o}	o̲
grave	\`{o}	ò			
dotless i	\i	ı	dotless j	\j	ȷ
	\'{\i}	í		\v{\j}	ǰ

Table 1.3: European accents.

Name	Type	Typeset	Type	Typeset
a-ring	\aa	å	\AA	Å
aesc	\ae	æ	\AE	Æ
ethel	\oe	œ	\OE	Œ
eszett	\ss	ß	\SS	SS
inverted question mark	?`	¿		
inverted exclamation mark	!`	¡		
slashed L	\l	ł	\L	Ł
slashed O	\o	ø	\O	Ø

Table 1.4: European characters.

Name	Type	Typeset
ampersand	\&	&
asterisk bullet	\textasteriskcentered	*
backslash	\textbackslash	\
bar (caesura)	\textbar	|
brace left	\{	{
brace right	\}	}
bullet	\textbullet	•
circled a	\textcircled{a}	ⓐ
circumflex	\textasciicircum	^
copyright	\copyright	©
dagger	\dag	†
double dagger (diesis)	\ddag	‡
dollar	\$	$
double quotation left	\textquotedblleft or ``	“
double quotation right	\textquotedblright or ''	”
em dash	\textemdash or ---	—
en dash	\textendash or --	–
exclamation down	\textexclamdown or !`	¡
greater than	\textgreater	>
less than	\textless	<
lowline	_	_
midpoint	\textperiodcentered	·
octothorp	\#	#
percent	\%	%
pilcrow (paragraph)	\P	¶
question down	\textquestiondown or ?`	¿
registered trademark	\textregistered	®
section	\S	§

Table 1.5: Extra text symbols.

1.2.8 Logos and dates:

TeX produces TEX, *LaTeX* produces LATEX, and *LaTeXe* produces LATEXe (the original name of the current version of LATEX). The *AmS* command produces the logo *AMS*.

Remember to type \TeX\␣ or \TeX{} if we need a space after TEX (similarly for the others).

- *time* is the time of day in minutes since midnight
- *day* is the day of the month
- *month* is the month of the year
- *year* is the current year

We can include these numbers in our document by using the *the* command:

Year: *the**year*; month: *the**month*; day: *the**day*

produces a result such as

Year: 2015; month: 7; day: 11

Of more interest is the *today* command, which produces today's date in the form: July 11, 2015.

1.3 Comments and footnotes:

Various parts of your source file do not get typeset like most of the rest. The two primary examples are comments that do not get typeset at all and footnotes that get typeset at the bottom of the page.

1.3.1 Comments:

The % symbol tells LATEX to ignore the rest of the line. A common use might be a comment to yourself in the source file:

therefore, a reference to Theorem˜1 % check this!

The % symbol has many uses. For instance, a document class command,

\documentclass[twocolumn,twoside,legalpaper]{amsart}

may be typed with explanations, as

\documentclass[%

twocolumn,% option for two-column pages
twoside,% format for two-sided printing
legalpaper% print on legal-size paper
]{amsart}

so we can easily comment out some at a later time, as in

\documentclass[%
twocolumn,% option for two-column pages
twoside,% format for two-sided printing
legalpaper% print on legal-size paper
]{amsart}

Notice that the first line is terminated with a % to comment out the end-of-line character.

Some command arguments do not allow any spaces. If we want to break a line within an argument list, we can terminate the line with a %, as shown in the previous example.

It is often useful to start a document with a comment line giving the file name and identifying the earliest version of LATEX that must be used to typeset it.

%This is article.tex
\NeedsTeXFormat{LaTeX2e}[1994/12/01]

The second line specifies the December 1, 1994 (or later) release of LATEX. We may need to use such a declaration if your document uses a feature that was not available in earlier releases.

Note that % does not comment out lines in a BIBTEX database document.

Symbolic referencing:

With every *\label* command I add the commented-out form of the symbolic reference.

So if we start a new theorem, we type *\the* and my text expander inserts the following in the article:

\begin{theorem}\label{T:xx}
%Theorem~\ref{T:xx}
\end{theorem}

The 25% rule:

If we want a % sign in text, make sure you type it as \%. Otherwise, % comments out the rest of the line. LATEX does not produce a warning.

Using % to comment out large blocks of text can be tedious even with block comment. The verbatim package includes the comment environment:

\begin{comment}
...the commented out text...
\end{comment}

Rule: comment **environments:**

1. \end{comment} must be on a line by itself.

2. There can be no comment within a comment.

In other words,

\begin{comment}
commented out text...
\begin{comment}
some more commented out text...
\end{comment}
and some more commented out text...
\end{comment}

is not allowed. LATEX may give one of several messages, depending on the circumstances. For instance,

! LaTeX Error: *\begin{document} ended by \end{comment}.*

l.175 *\end{comment}*

Locating errors:

The comment environment can be very useful in locating errors.

Suppose we have unbalanced braces in our source file. Working with a *copy* of our source file, comment out the first half at a safe point (not within an environment!) and typeset. If we still get the same message, the error is in the second half. If there is no error message, the error is in the first half. Comment out the half that has no error.

Now comment out half of the remaining text and typeset again. Check to see whether the error appears in the first half of the remaining text or the second. Continue applying this method until we narrow down the error to a paragraph that we can inspect visually.

Since the comment environment requires the verbatim package, we must include the line

\usepackage{verbatim}

in the preamble of the source file.

1.3.2 Footnotes:

The text of a footnote is typed as the argument of a *\footnote* command. This footnote is typed as

\footnote{Footnotes are easy to place.}

If you want to use symbols to designate the footnotes, instead of numbers, type the command

\renewcommand{\thefootnote}

{\ensuremath{\fnsymbol{footnote}}}

before the first footnote; this provides up to nine symbols.

In addition, there are title-page footnotes, such as the *\thanks* and \date commands in the top matter.

We can add a footnote marked by * to the title of an article. For instance, type the title

\title[Complete congruence lattices]%

{Complete congruence lattices*}

and add the lines

{\renewcommand{\thefootnote}{\fnsymbol{footnote}}
\setcounter{footnote}{1}
\footnotetext{Lecture delivered at the \AMS
annual meeting in Brandon.}
\setcounter{footnote}{0}
}

The footnote will appear as the first footnote on page 1 marked by ∗. All the other footnotes are unmarked.

1.4 Changing font characteristics:

Although a document class and its options determine how LATEX typesets characters, there are occasions when you want control over the shape or size of the font used.

1.4.1 Basic font characteristics:

We do not have to be a typesetting expert to recognize the following basic font attributes:

Shape:

Normal text is typeset:

upright (or *roman*) as this text

slanted *as this text*

italic	*as this text*
small caps	AS THIS TEXT

Monospaced and proportional:

Typewriters use *monospaced* fonts, that is, fonts all of whose characters are of the same width. Most text editors display text using a monospaced font. LATEX calls monospaced fonts *typewriter style*. The normal text is typeset in a *proportional* font, such as *"proportional text with ii and mm"*, in which ii is narrow and mm is wide:

Monospaced and proportional Typewriters use *monospaced* fonts,

mmmmmm
iiiiii
} monospaced

mmmmmm
iiiiii
} proportional

Serifs:

A *serif* is a small horizontal (sometimes vertical) stroke used to finish off a vertical stroke of a letter, as on the top and bottom of the letter M. LATEX's standard serif font is Computer Modern roman, such as "serif text". Fonts without serifs are called *sans serif*, such as "sans serif text". Sans serif fonts are often used for titles or for special emphasis.

Series (weight and width):

The *series* is the combination of weight and width. A font's *weight* is the thickness of the strokes and the *width* is how wide the characters are.

Light, *medium* (or *normal*), and *bold* often describe weight.

Narrow (or *condensed*), *medium* (or *normal*), and *extended* often describe width. The Computer Modern family includes **bold fonts**. Traditionally, when the user asks for bold CMfonts, LATEX actually provides *bold extended* (a somewhat wider version).

Size:

Most LATEX articles are typeset with 10 point text unless otherwise instructed. Larger sizes are used for titles, section titles, and so on. Abstracts and footnotes are often set in 8-point type.

Font family:

The collections of all sizes of a font is called a *font family*.

1.4.2 Document font families:

In a document class, the style designer designates three document font families:

1. *Roman* (upright and serifed) document font family
2. *Sans serif* document font family
3. *Typewriter style* document font family

and picks one of these (for articles, as a rule, the roman document font family) as the *document font family* or *normal family*. In all the examples, the document font family is the roman document font family except for presentations which use sans serif. When we use Computer Modern fonts in LATEX, which is the default, the three document font families are Computer Modern roman, Computer Modern sans serif, and Computer Modern typewriter. The document font family is Computer Modern roman.

The roman document font family is Times, the sans serif document font family is Helvetica, and the typewriter style document font family is Computer Modern typewriter. The document font family is the roman document font family Times.

The document font family (normal family) is the default font. We can always switch back to it with

\textnormal{...}

or

{\normalfont ...}

Table 1.6 shows these two commands and three additional pairs of commands to help we switch among the three basic document font families. It also shows the command pairs for the basic font shapes.

Command with Argument	Command Declaration	Switches to the font family
\textnormal{...}	{\normalfont ...}	document
\emph{...}	{\em ...}	*emphasis*
\textrm{...}	{\rmfamily ...}	roman
\textsf{...}	{\sffamily ...}	sans serif
\texttt{...}	{\ttfamily ...}	typewriter style
\textup{...}	{\upshape ...}	upright shape
\textit{...}	{\itshape ...}	*italic shape*
\textsl{...}	{\slshape ...}	*slanted shape*
\textsc{...}	{\scshape ...}	SMALL CAPITALS
\textbf{...}	{\bfseries ...}	**bold**
\textmd{...}	{\mdseries ...}	normal weight and width

Table 1.6: Font family switching commands.

The font-changing commands of Table 1.6 come in two forms:

- A command with an argument, such as *\textrm{...}*, changes its argument. These are short commands, i.e., they cannot contain a blank line or a *\par* command.
- A command declaration, such as *\rmfamily*, carries out the font change following the command and within its scope

We should always use commands with arguments for small changes within a paragraph. They have two advantages:

- We are less likely to forget to change back to the normal font.
- We do not have to worry about italic corrections.

Note that *MakeIndex* requires we to use commands with arguments to change the font in which page numbers are typeset.

For font changes involving more than one paragraph, use command declarations.

These commands are preferred if we want to create custom commands and environments.

1.4.3 Shape commands:

There are five pairs of commands to change the font shape:

- *\textup{...}* or *{\upshape ...}* switch to the upright shape.
- *\textit{...}* or *{\itshape ...}* switch to the *italic shape*.
- *\textsl{...}* or *{\slshape ...}* switch to the *slanted shape*.
- *\textsc{...}* or *{\scshape ...}* switch to small capitals.
- *\emph{...}* or *{\em ...}* switch to *emphasis*.

The document class specifies how emphasis is typeset. As a rule, it is italic or slanted unless the surrounding text is italic or slanted, in which case it is upright. For instance,

\emph{Rubin space}

in the statement of a theorem is typeset as

the space satisfies all three conditions, a so-called Rubin space *that ...*

The emphasis changed the style of Rubin space from italic to upright.

Abbreviations and acronyms:

For abbreviations and acronyms use small caps, except for two-letter geographical acronyms.

So Submitted to TUG should be typed as

Submitted to *\textsc{tug}*

Note that only the lowercase characters in the argument of the *\textsc* command are printed as small caps.

1.4.4 Italic corrections:

The phrase

when using a *serif* font

may be typed as follows:

when using a *{\itshape serif\/}* font

The \/ command before the closing brace is called an *italic correction.* Notice that *{\itshape M}*M typesets as *M*M, where the *M* is leaning into the M. Type {\itshape M\/}M to get the correct spacing *M*M. Compare the typeset phrase from the previous example with and without an italic correction:

when using a *serif* font

when using a *serif* font

The latter is not as pleasing to the eye.

Italic correction:

Rule 1:

If the emphasized text is followed by a period or comma, we should not type the italic correction.

For example,

Do not forget. My party is on Monday.

should be typed as

{*itshape* Do not forget.} My party is on Monday.

Rule 2:

The shape commands with arguments do not require italic correction. The corrections are provided automatically where needed.

Thus, we can type the phrase when using a *serif* font the easy way:

when using a *\textit{serif}* font

Whenever possible, let LATEX take care of the italic correction. However, if LATEX is adding an italic correction where we feel it is not needed, we can override the correction with the *\nocorr* command. LATEX does not add an italic correction before a period or a comma. These two punctuation marks are stored in the *\nocorrlist* command.

By redefining this command, we can modify LATEX's behavior.

Rule 3:

The italic correction is required with the commands *\itshape, \slshape, \em.*

1.4.5 Series:

These attributes play a very limited role with the Computer Modern fonts. There is only one important pair of commands,

\textbf{...} {\bfseries ...}

to change the font to bold (actually, bold extended). The commands

\textmd{...} {\mdseries ...}

which set both the weight and width to medium (normal) are seldom needed.

1.4.6 Size changes:

Standard LATEX documents are typeset in 10 point type. The 11 point and 12 point type are often used for better readability and some journals require 12 point—if this is the case, use the 12pt document class option. The 8pt and 9pt document class options are rarely used. The sizes of titles, subscripts, and superscripts are automatically set by the document class, in accordance with the font size option.

If we must change the font size for some text—it is seldom necessary to do so in an article—the following command declarations are provided:

\Tiny *\tiny* *\SMALL* *\Small* *\small*

\normalsize

\large *\Large* *\LARGE* *\huge* *\Huge*

The command \SMALL is also called *scriptsize* and the command *Small* is also called *footnotesize*. The font size commands are listed in order of increasing—to be more precise, nondecreasing—size.

Command	Sample text
\Tiny	sample text
\tiny	sample text
\SMALL or \scriptsize	sample text
\Small or \footnotesize	sample text
\small	sample text
\normalsize	sample text
\large	sample text
\Large	sample text
\LARGE	sample text
\huge	sample text
\Huge	sample text

Table 1.7: Font size commands.

Two commands allow the user to increase or decrease font size: *larger* moves up one size, *smaller* moves down one. Both commands take an optional argument. For example, *larger[2]* moves up 2 sizes.

1.4.7 Orthogonality:

We are now familiar with the commands that change the font family, shape, series, and size. Each of these commands affects one and only one font attribute. For example, if we change the series, then the font family, shape, and size do not change. These commands act independently. In LATEX terminology, the commands are *orthogonal.* From the user's point of view this behavior has an important consequence: *The order in which these commands are given does not matter.* Thus

\Large \itshape \bfseries

has the same effect as

\bfseries \itshape \Large

Note that LATEX 2.09's two-letter commands are not orthogonal.

Orthogonality also means that we can combine these font attributes in any way we like. For instance, the commands

\sffamily \slshape \bfseries \Large

instruct LATEX to change the font family to sans serif, the shape to slanted, the series to bold, and the size to *\Large*.

1.4.8 Obsolete two-letter commands:

Users of LATEX 2.09 and AMS-LATEX version 1.1 are accustomed to using the two-letter commands *\bf*, *\it*, *\rm*, *\sc*, *\sf*, *\sl*, and *\tt*. These commands are not part of LATEX. They are, however, still defined in most document classes. The two-letter commands

1. switch to the document font family,
2. change to the requested shape.

There are a number of reasons not to use them. The two-letter commands

- are not part of LATEX,
- require manual italic corrections,
- are not orthogonal.

\slshape \bfseries is the same as *\bfseries \slshape* (slanted bold), but *\sl\bf* is not the same as *\bf\sl*. Indeed, *{\sl\bf sample}* gives **sample** and *{\bf\sl sample}* produces *sample*.

1.4.9 Low-level commands:

The font-characteristic changing commands we have discussed so far in this section are the *high-level* font commands. Each of these commands is implemented by LATEX and the document class using *low-level* font commands. The low-level commands have been developed for document class and package writers.

There is one use of low-level commands we should keep in mind. When we choose a font size for our document or for some part thereof, we also determine the

baselineskip, the distance from the baseline of one line to the baseline of the next. Typically, a 10-point font size uses a 12 point *baselineskip*. Occasionally, we may want to change the font size along with the *baselineskip*. A command for accomplishing this is

\fontsize{9pt}{11pt}\selectfont

which changes the font size to 9 point and the *baselineskip* to 11 point. To make this change for a single paragraph, we can type

{%special paragraph
\fontsize{9pt}{11pt}\selectfont

text

}%end special paragraph

Observe the blank line that follows text and marks the end of the paragraph; *par* would accomplish the same thing.

1.5 Lines, paragraphs, and pages

When typesetting a document, LATEX breaks the text into lines, paragraphs, and pages. Sometimes we may not like how LATEX has chosen to lay out your text. There are ways to influence how LATEX does its work and these are discussed in this section.

1.5.1 Lines:

LATEX typesets a document one paragraph at a time. It tries to split the paragraph into lines of equal width. If it fails to do so successfully and a line is too wide, we get an overfull \hbox message. Here is a typical example:

Overfull \hbox (15.38948pt too wide) in paragraph
at lines 11—16

[]\OT1/cmr/m/n/10 In sev-eral sec-tions of the course

on ma-trix

the-ory, the strange term ‘‘hamiltonian-

The log file records these messages. To place a visual warning in the typeset version of our document as well, use the draft document class option

\documentclass[draft]{amsart}

Lines that are too wide are be marked with a *slug* (a black box) in the margin. A slug is a vertical bar of width \overfullrule.

Do not worry about such messages while writing the document. If we are preparing the final version and receive a message for an overfull \hbox, the first line of defense is to see whether optional hyphens would help. Read the warning message carefully to see which words LATEX cannot hyphenate properly. If adding optional hyphens does not help, a simple rephrasing of the problem sentence often does the trick.

Recall that there are 72.27 points in an inch. So if the message indicates a 1.55812 pt overflow, for instance, we can safely ignore it.

Breaking lines:

There are two forms of the line breaking command:

- The \\ and \newline commands break the line at the point of insertion but do not stretch it.
- The \linebreak command breaks the line at the point of insertion and stretches the line to make it of the normal width.

The text following any of these commands starts at the beginning of the next line, without indentation. The \\ command is often used, but \linebreak is rarely seen. The effect of these commands:

There are two forms of the line breaking command:

There are two forms\\ of the line breaking command:

There are two forms \newline of the line breaking command:

There are two forms \linebreak of the line breaking command:

typeset as

There are two forms of the line breaking command:
There are two forms
of the line breaking command:
There are two forms
of the line breaking command:
There are two forms of the line breaking command:

If we force a line break in the middle of a paragraph with the \linebreak command and LATEX thinks that there is too little text left on the line to stretch it to full width, we get a message such as

Underfull \hbox (badness 4328) in paragraph
at lines 8--12

The \\ command has two important variants:

- \\[*length*], where *length* is the interline space we wish to specify after the line break, for instance, 12pt, .5in, or 1.2cm. Note how the units are abbreviated.
- *, which prohibits a page break following the line break.

The *[*length*] form combines the two variants. We illustrate the \\[*length*] command:

It is also semimodular.\\[15pt]
In particular,

which is typeset as

It is also semimodular.
In particular,

Since \\ can be modified by * or by[ ], LATEX may get confused if the line after a \\ command starts with a * or [. In such cases, type * as {*} or [as {[}. For instance, to get

There are two sources of problems:
{[}a] The next line starts with \texttt{[}

There are two sources of problems:\\

{[}a] The next line starts with \texttt{[}.

If we fail to type {[}, you get the message
! Missing number, treated as zero.
<to be read again>
a
l.16 [a]
The next line starts with \texttt{[}.

Rule: \\

Without optional arguments, the \\ command and the \newline command are the same *in text,* but not within environments or command arguments. We can qualify the \linebreak command with an optional argument: 0 to 4. The higher the argument, the more it forces the occurrence of a line break. \linebreak[4] is the same as \linebreak, while \linebreak[0] allows the line break but does not force it.

The \nolinebreak command plays the opposite role. \nolinebreak[0] = \linebreak[0], and \nolinebreak[4] = \nolinebreak. \nolinebreak is seldom used since the tie (~) and the \text command accomplish the same goal most of the time.

Double spacing:

It is convenient to proofread documents double spaced. Some journals even require submissions to be double spaced.

To typeset a document double spaced, include the command

\renewcommand{\baselinestretch}{1.5}

in its preamble.

Alternatively, use George D. Greenwade's setspace. Load this package with a

\usepackage{setspace}

command in the preamble of the document and specify

\doublespacing

in the preamble. This changes not just the line spacing but a number of other parameters to make our article look good.

1.5.2 Paragraphs:

Paragraphs are separated by blank lines or by the \par command. LATEX error messages always show paragraph breaks as \par. The \par form is also very useful in custom commands and environments.

In some document classes, the first line of a paragraph is automatically indented. Indentation can be prevented with the \noindent command and can be forced with the \indent command.

Sometimes—for instance, in a schedule, glossary, or index—we may want a *hanging indent*, where the first line of a paragraph is not indented, and all the others are indented by a specified amount.

Hanging indents are created by specifying the amount of indentation specified by \hangindent and set with the \setlength command:

\setlength{\hangindent}{30pt}
\noindent
\textbf{sentence} a group of words terminated by
a period, exclamation point, or question mark.
\setlength{\hangindent}{30pt}
\noindent
\textbf{paragraph} a group of sentences terminated by a
blank line or by the new paragraph command.

produces

sentence a group of words terminated by a period, exclamation point, or question mark.

paragraph a group of sentences terminated by a blank line or by the new paragraph command.

Notice that the \setlength command must be repeated for each paragraph. Sometimes we may want to change the value of \hangafter, the length command that specifies the number of lines not to be indented. The default value is 1.
To change it to 2, use the command

\setlength{\hangafter}{2}

1.5.3 Pages:

There are two page breaking commands:

- \newpage, which breaks the page at the point of insertion but does not stretch the content
- \pagebreak, which breaks the page at the point of insertion and stretches the page's content to normal length

Text following either command starts at the beginning of the next page, indented. The page breaking commands are analogous to the line breaking commands. This analogy continues with the optional argument, 0 to 4:

\pagebreak[0] to \pagebreak[4]
\nopagebreak[0] to \nopagebreak[4]

There are also special commands for allowing or forbidding page breaks in multiline math displays.

When preparing the final version of a document, we may have to extend or shrink a page by a line or two to prevent it from breaking at an unsuitable line. We can do so with the \enlargethispage command. For instance,

\enlargethispage{\baselineskip}

adds one line to the page length. On the other hand,

enlargethispage{-\baselineskip}

makes the page one line shorter.

enlargethispage{10000pt}

makes the page very long.

The *-ed version, \enlargethispage*, squeezes the page as much as possible.

There are two more variants of the \newpage command. The

clearpage

command does a \newpage and typesets all the figures and tables waiting to be processed. The variant

cleardoublepage

is used with the two-side document class option. It does a \clearpage and in addition makes the next printed page a right-hand, that is, odd numbered, page, by inserting a blank page if necessary. If for our document class this does not work, use the package *cleardoublepage.sty* in the samples folder.

1.5.4 Multicolumn printing:

Many document classes provide the two-column option for two-column typesetting. In addition, there is a \twocolumn command which starts a new page by issuing a \clearpage and then typesets in two columns. An optional argument provides a two-column wide title. Use the \onecolumn command to switch back to a one-column format.

Frank Mittelbach's *multicol package* provides the much more sophisticated multicols environment, which can start in the middle of a page, can handle more than two columns, and can be customized in a number of ways.

1.6 Spaces:

The judicious use of horizontal and vertical space is an important part of the formatting of a document. Fortunately, most of the spacing decisions are made by the document class, but LATEX has a large number of commands that allow the user to insert horizontal and vertical spacing.

Remember that LATEX ignores excess spaces, tabs, and end-of-line characters. If we need to add horizontal or vertical space, then we must choose from the commands in this section.

1.6.1 Horizontal spaces:

In this section, we discuss fixed length horizontal space commands.

When typing text, there are three commands that are often used to create horizontal space, shown between the bars in the display below:

```
\␣          ␣
\quad       ␣␣
\qquad      ␣␣␣␣
```

The \quad command creates a 1 em space and \qquad creates a 2 em space. The interword space created by \␣ can both stretch and shrink. There are other commands that create smaller amounts of space.

The \hspace command takes a length as a parameter. The length may be negative.

For example,

```
\textbar\hspace{12pt}\textbar      |  |
\textbar\hspace{.5in}\textbar      |      |
\textbar\hspace{1.5cm}\textbar     |          |
```

or *hspace{-40pt}*. The command \hspace is often used with a negative argument when placing illustrations.

The command produces a space the width and height of the space that would be occupied by its typeset argument

\textbar need space\textbar	*|need space|*
\textbar\textbar	| |

and

alpha gamma
 beta delta

produces

alpha gamma
beta delta

The \phantom command is very useful for fine tuning aligned math formulas. The variant \hphantom{*argument* } creates a space with the horizontal dimension that would be occupied by its typeset *argument* and with zero height.

Horizontal space variant:

When LATEX typesets a line, it removes all spaces from the beginning of the line, including the space created by \hspace, \quad, and other spacing commands. Using the *-ed variant of \hspace, \hspace*, prevents LATEX from removing the space we have specified.
For example,

And text
\hspace{20pt}And text
\hspace{20pt}And text*

is typeset as

And text
And text
And text

Use the \hspace* command for creating customized indentation. To indent a paragraph by 24 points, give the command

\noindent\hspace{24pt}And text*

which typesets as

And text

To break a line and indent the next line by 24 points, give the command

And text\\
\hspace{24pt}And text*

which produces

And text
And text

1.6.2 Vertical spaces:

We can add some interline space with the command \\[*length*]. We can also do it with the \vspace command, which works just like the \hspace command, except that it creates vertical space. Here are some examples:

\vspace{12pt} \vspace{.5in} \vspace{1.5cm}.

Standard amounts of vertical space are provided by the three commands

\smallskip \medskip \bigskip

The space these commands create depends on the document class and the font size.

Rule for vertical space commands:

All vertical space commands add the vertical space *after* the typeset line in which the command appears.

To obtain

end of text.

New paragraph after vertical space

type

end of text.
\vspace{12pt}
New paragraph after vertical space

The following example illustrates the unexpected way the vertical space is placed if the command that creates it does not start a new paragraph:

end of text.
\vspace{12pt}
The following example illustrates the unexpected way
the vertical space is placed if the
command that creates it does not start a new paragraph:

It typesets as

end of text. The following example illustrates the unexpected way the vertical

space is placed if the command that creates it does not start a new paragraph:

Vertical space variants:

LATEX removes vertical space from the beginning and end of each page, including space produced by \vspace. The space created by the variant \vspace* is not removed by LATEX under any circumstances. Use this command, for instance, to start the typeset text (say, of a letter) not at the top of the page.

The \phantom command has also a vertical variant: \vphantom. The command \vphantom{*argument* } creates a vertical space with the vertical dimension that would be occupied by its typeset argument, *argument*.

1.6.3 Relative spaces:

The length of a space is usually given in *absolute units:* 12pt (points), .5cm (centimeters), 1.5in (inches). Sometimes, *relative units*, em and ex, are more appropriate, units that are relative to the size of the letters in the current font. The unit 1 em is approximately the width of an M in the current font, while 1 ex is approximately the height of an x in the current font. These units are used in commands such as

\hspace{12em} and \vspace{12ex}

The \quad and \qquad commands produce 1 em and 2 em spaces.

1.6.4 Expanding spaces:

Horizontal spaces:

The \hfill, \dotfill, and \hrulefill commands fill all available space in the line with spaces, dots, or a horizontal line, respectively. If there are two of these commands on the same line, the space is divided equally between them. These commands can be used to center text, to fill lines with dots in a table of contents, and so on.

To obtain

2. Boxes. 34

ABC and DEF

ABC and DEF

type

2. Boxes\dotfill 34
ABC\hfill and\hfill DEF
ABC\hrulefill and\hrulefill DEF

In a centered environment—such as a \title or a center environment we can use \hfill to set a line flush right:

This is the title

First Draft

Author

To achieve this effect, type

\begin{center}
This is the title\\
\hfill First Draft\\
Author
\end{center}

Vertical spaces:

The vertical analogue of \hfill is \vfill. This command fills the page with vertical space so that the text before the command and the text after the command stretch to the upper and lower margin.

The command \vfill stands for \vspace{\fill}, so it is ignored at the beginning of a page. Use \vspace*{\fill} if we need it at the beginning of a page.

1.7 Boxes:

Sometimes it can be useful to typeset text in an imaginary box, and treat that box as a single large character. A single-line box can be created with the \text or \makebox commands and a multiline box of a prescribed width can be created with the \parbox command or minipage environment.

1.7.1 Line boxes:

The \text command provides a *line box* that typesets its argument without line breaks. As a result, we may find the argument extending into the margin. The resulting box is handled by LATEX as if it were a single large character. For instance,

\text{database}

causes LATEX to treat the eight characters of the word database as if they were one.

This technique has a number of uses.

The argument of \text is typeset in a size appropriate for its use, for example, as a subscript or superscript.

Line boxes—a refinement:

The \mbox command is the short form of the \makebox command. Both \mbox and \text prevent breaking the argument, but \mbox does not change size in subscripts and superscripts.

The full form of the \makebox command is

makebox[width][alignment]{text }

where the arguments are

- *width,* the (optional) width of the box. If [*width*] is omitted, the box is as wide as necessary to enclose its contents.
- *alignment,* (optionally) one of c (the default), l, r, or s. The text is centered by default, l sets the argument flush left, r right, and s stretches the text the full length of the box if there is blank space in the argument.
- *text,* the text in the box.

A *width* argument can be specified in inches (in), centimeters (cm), points (pt), em, or ex.

The following examples,

\makebox{Short title.}End
\makebox[2in][l]{Short title.}End
\makebox[2in]{Short title.}End
\makebox[2in][r]{Short title.}End
\makebox[2in][s]{Short title.}End

typeset as

Short title.End

Short title. *End*

Short title. *End*

Short title.End

Short *title.End*

The optional width argument, *width*, can use four length commands:

\height \depth \totalheight \width

These are the dimensions of the box that would be produced without the optional width argument.

Here is a simple example. The command

\makebox{hello}

makes a box of width \width. To typeset hello in a box three times the width, that is, in a box of width 3\width, use the command

\makebox[3\width]{hello}

So

start\makebox[3\width]{hello}end

typesets as

start hello end

The formal definition of these four length commands is the following:

- \height is the height of the box above the baseline
- \depth is the depth of the box below the baseline
- \totalheight is the sum of \height and \depth
- \width is the width of the box

1.7.2 Frame boxes:

Boxed text is very emphatic. For example, Do not touch! is typed as

\fbox{Do not touch!}

This is a *frame box*, hence the command \fbox or \framebox.

Boxed text cannot be broken, so if we want a frame around more than one line of text, we should put the text as the argument of a \parbox command or within a minipage environment, and then put that into the argument of an \fbox command. For instance,

\fbox{\parbox{3in}{Boxed text cannot be broken,
so if we want to frame more than one line
of text, place it in the argument of a
\bsl\texttt{parbox}
command or within a
\texttt{minipage} environment.}}

produces

Boxed text cannot be broken, so if we want to frame more than one line of text, place it in the argument of a \parbox command or within a minipage environment.

The \framebox command works exactly like \makebox, except that it draws a frame around the box.

\framebox[2in][l]{Short title}

produces

Short title

We can use this command to typeset the number 1 in a square box, as required by the title of Michael Doob's [12]:

TEX *Starting from* 1

\framebox{\makebox[\totalheight]{1}}

which typesets as

1

Note that

\framebox[\totalheight]{1}

typesets as

1

which is not a square box. Indeed, \totalheight is the height of 1, which becomes the width of the box. The total height of the box, however, is the height of the character 1 to which we have to add twice the \fboxsep, the separation between the contents of the box and the frame, defined as 3 points, and twice the \fboxrule, the width of the line, or rule, defined as 0.4 points. These lengths are in general also added to the width of the box, but not in this case, because we forced the width to equal the height of the character.

We can use the \fbox command to frame the name of an author:

\author{\fbox{author's name}}

1.7.3 Paragraph boxes:

A paragraph box works like a paragraph. The text it contains is wrapped around into lines. The width of these lines is set by the user.

The \parbox command typesets the contents of its second argument as a paragraph with a line width supplied as the first argument. The resulting box is handled by LATEX as a single large character. For example, to create a 3-inch wide column,

Fred Wehrung's new result shows the limitation of E. T. Schmidt's construction, especially for large lattices.

type

\parbox{3in}{Fred Wehrung's new result shows the limitation of E.\,T. Schmidt's construction, especially for large lattices.}

Paragraph boxes are especially useful when working within a tabular environment.

The width of the paragraph box can be specified in inches (in), centimeters (cm), points (pt), or the relative measurements em and ex, among others.

1.7.4 Marginal comments:

A variant of the paragraph box, the \marginpar command, allows us to add marginal comments. So

\marginpar{Do not use this often}

produces the comment displayed in the margin.

Rule for Marginal comments and math environments:

Do not use marginal comments in equations or multiline math environments.

Avoid using too many marginal comments on any given page—LATEX may have to place some of them on the next page.

If the document is typeset two-sided, then the marginal comments are set in the outside margin. The form

\marginpar[left-comment]{right-comment }

uses the required argument *right-comment* when the marginal comment is set in the right margin and the optional argument *left-comment* when the marginal comment is set in the left margin.

The width of the paragraph box for marginal comments is stored in the length command \marginparwidth. If we want to change it, use

\setlength{\marginparwidth}{*new_width* }

as in

\setlength{\marginparwidth}{90pt}

The default value of this width is set by the document class.

1.7.5 Solid boxes:

A solid filled box is created with a \rule command. The first argument is the width and the second is the height. For instance, to obtain

end of proof symbol:■

type

end of proof symbol: \rule{1.6ex}{1.6ex}

In fact, this symbol is usually slightly lowered:

end of proof symbol: ■

This positioning is done with an optional first argument:

end of proof symbol: \rule[-.23ex]{1.6ex}{1.6ex}

Here is an example combining \rule with \makebox and \hrulefill:

1 inch:\quad\makebox[1in]{\rule{.4pt}{4pt}%
\hrulefill\rule{.4pt}{4pt}}

which produces

1 inch: ⌊————————⌋

Struts:

Solid boxes of zero width are called *struts*. Struts are invisible, but they force LATEX to make room for them, changing the vertical alignment of lines. Standard struts can also be added with the \strut or \mathstrut command. To see how struts work, compare

$\boxed{\text{ab}}$ and $\boxed{\text{ab}}$ and $\boxed{\text{ab}}$

typed as

\fbox{ab} and \fbox{\strut ab} and \fbox{$\mathstrut$ab}

Struts are especially useful for fine tuning tables and formulas.

Zero distance:

0pt, 0in, 0cm, 0em all stand for zero width. 0 by itself is not acceptable. For example, *\rule{0}{1.6ex}* gives the message

! Illegal unit of measure (pt inserted).
<to be read again>
h
l.251 \rule{0}{1.6ex}

If the \rule command has no argument or only one, LATEX generates a message.

1.7.6 Fine tuning boxes:

The command

\raisebox{*displacement* }{*text* }

typesets *text* in a box with a vertical *displacement*. If *displacement* is positive, the box is raised; if it is negative, the box is lowered.
The \raisebox command allows us to play games:

fine-\raisebox{.5ex}{tun}\raisebox{-.5ex}{ing}

produces fine-$^{\text{tun}}_{\text{ing}}$.

The \raisebox command has two optional arguments:

\raisebox{0ex}[1.5ex][0.75ex]{text }

forces LATEX to typeset *text* as if it extended 1.5 ex above and 0.75 ex below the line, resulting in a change in the interline space above and below the line.

Unit-II:

Text environments: some general rules for displayed text environments - List of environments-style and size environments - proclamations (theorem-like structures) - Proof environments - Tabular environments - Tabbing environments - Miscellaneous displayed text environments.

2.0 Text environments:

Introduction:

There are three types of text environments in LATEX:

1. Displayed text environments; text within such an environment usually is typeset with some vertical space around it
2. Text environments that create a “large symbol”
3. Style and size environments.

2.1 Some general rules for displayed text environments:

As we know, blank lines play a special role in LATEX, usually indicating a paragraph break. Since displayed text environments structure the printed display themselves, the rules about blank lines are relaxed somewhat. However, a blank line trailing an environment signifies a new paragraph for the text following the environment.

Blank lines in displayed text environments:

1. Blank lines are ignored immediately after \begin{*name* } or immediately before \end{*name* } except in a verbatim environment.
2. A blank line after \end{*name* } forces the text that follows to start a new paragraph.
3. As a rule, we should not have a blank line before \begin{*name* }.
4. The line after any theorem or proof always begins a new paragraph, even if there is no blank line or \par command.

2.2 List environments:

LATEX provides three list environments: enumerate, itemize, and description. LATEX also provides a generic list environment that can be customized to fit your needs.

Most document classes redefine the spacing and some stylistic details of lists, especially since the list environments in the legacy document classes are not very pleasing. In this section, the list environments are formatted as they are by our standard document class, *amsart*.

2.2.1 Numbered lists:

A *numbered list* is created with the enumerate environment:

This space has the following properties:

(1) Grade 2 Cantor;
(2) Half-smooth Hausdorff;
(3) Metrizably smooth.

Therefore, we can apply the Main Theorem.

typed as

```
\noindent This space has the following properties:
\begin{enumerate}
   \item Grade 2 Cantor\label{Cantor};
   \item Half-smooth Hausdorff\label{Hausdorff};
   \item Metrizably smooth\label{smooth}.
\end{enumerate}
Therefore, we can apply the Main Theorem.
```

Each item is introduced with an \item command. The numbers LATEX generates can be labeled and cross-referenced. This construct can be used in theorems and definitions, for listing conditions or conclusions.

If we use \item in the form \item[], we get an unnumbered item in the list, while \item[a] replaces the number of the item with a. This is another form of absolute referencing.

2.2.2 Bulleted lists:

A *bulleted list* is created with the itemize environment:

We set out to accomplish a variety of goals:

- To introduce the concept of smooth functions.
- To show their usefulness in differentiation.
- To point out the efficacy of using smooth functions in Calculus.

is typed as

```
\noindent We set out to accomplish a variety of goals:
\begin{itemize}
   \item To introduce the concept of smooth functions.
   \item To show their usefulness in differentiation.
   \item To point out the efficacy of using smooth
      functions in Calculus.
\end{itemize}
```

2.2.3 Captioned lists:

In a *captioned list* each item has a title (caption) specified by the optional argument of the \item command. Such lists are created with the description environment:

In this introduction, we describe the basic techniques:

Chopped lattice: a reduced form of a lattice;
Boolean triples: a powerful lattice construction;
Cubic extension: a subdirect power flattening the congruences.

is typed as

```
\noindent In this introduction, we describe
   the basic techniques:
\begin{description}
  \item[Chopped lattice] a reduced form of a lattice;
  \item[Boolean triples] a powerful lattice construction;
  \item[Cubic extensions] a subdirect power flattening
       the congruences.
\end{description}
```

2.2.4 A rule and combinations:

There is only one rule we must remember.

List environments:

An \item command must immediately follow

\begin{enumerate}, \begin{itemize}, or \begin{description}.

Of course, spaces and line breaks can separate them.

If we break this rule, we get a message. For instance,

```
\begin{description}
This is wrong!
  \item[Chopped lattice] a reduced lattice;
```

gives the message

```
! LaTeX Error: Something's wrong--perhaps a missing \item.

l.105   \item[Chopped lattice]
                              a reduced lattice;
```

If we see this message, remember the rule for list environments and check for text preceding the first \item.

We can nest up to four list environments; for instance,

(1) First item of Level 1.
    (a) First item of Level 2.
        (i) First item of Level 3.
            (A) First item of Level 4.
            (B) Second item of Level 4.
        (ii) Second item of Level 3.
    (b) Second item of Level 2.
(2) Second item of Level 1.

Referencing the second item of Level 4: 1(a)iB

which is typed as

```
\begin{enumerate}
   \item First item of Level 1.
   \begin{enumerate}
      \item First item of Level 2.
      \begin{enumerate}
         \item First item of Level 3.
         \begin{enumerate}
            \item First item of Level 4.
            \item Second item of Level 4.\label{level4}
          \end{enumerate}
          \item Second item of Level 3.
       \end{enumerate}
       \item Second item of Level 2.
   \end{enumerate}
   \item Second item of Level 1.
\end{enumerate}
Referencing the second item of Level 4: \ref{level4}
```

Note that the label level4 collected all four of the counters.

We can also mix list environments:

(1) First item of Level 1.
- First item of Level 2.
  (a) First item of Level 3.
    - First item of Level 4.
    - Second item of Level 4.
  (b) Second item of Level 3.
- Second item of Level 2.

(2) Second item of Level 1.

Referencing the second item of Level 4: 1a

which is typed as

```
\begin{enumerate}
   \item First item of Level 1.
   \begin{itemize}
      \item First item of Level 2.
      \begin{enumerate}
         \item First item of Level 3.
         \begin{itemize}
            \item First item of Level 4.
            \item Second item of Level 4.\label{enums}
          \end{itemize}
          \item Second item of Level 3.
       \end{enumerate}
       \item Second item of Level 2.
   \end{itemize}
   \item Second item of Level 1.
\end{enumerate}
Referencing the second item of Level 4: \ref{enums}
```

Now the label enums collects only the two enumerate counters.

The indentations are, of course, not needed. we use them to keep track of the level of nesting.

In all three types of list environment, the \item command may be followed by an optional argument, which is displayed at the beginning of the typeset item:

\item[label]

Note that for enumerate and itemize the resulting typography may leave something to be desired.

2.3 Style and size environments:

There are several text environments that allow us to set font characteristics. They have the same names as their corresponding command declarations:

```
rmfamily  sffamily  ttfamily
upshape  itshape  em  slshape  scshape
bfseries
```

For instance,

```
\begin{ttfamily}
   text
\end{ttfamily}
```

Horizontal alignment of a paragraph is controlled by the flushleft, flushright, and center environments. Within the flushright and center environments, it is customary to force new lines with the \\ command, while in the flushleft environment, we normally allow LATEX to wrap the lines.

These text environments can be used separately or in combination, as in

The **simplest**text environments set the printing style and size.
The commands and the environments have similar names.

typed as

```
\begin{flushright}
   The \begin{bfseries}simplest\end{bfseries}
   text environments set the
   printing style and size.\\
   The commands and the environments have similar names.
\end{flushright}
```

There are command declarations that correspond to these environments:

- \centering centers text
- \raggedright left aligns text
- \raggedleft right aligns text

The effect of one of these commands is almost the same as that of the corresponding environment except that the environment places additional vertical space before and after the displayed paragraphs. For such a command declaration to affect the way a paragraph is formatted, the scope must include the whole paragraph, including the blank line at the end of the paragraph, preferably indicated with a \par command.

The \centering command is used often with the \includegraphics command.

2.4 Proclamations (theorem-like structures):

Theorems, lemmas, definitions, and so forth are a major part of mathematical writing. In LATEX, these constructs are typed in displayed text environments called *proclamations* or *theorem*-like *structures*.

In the *firstarticle.tex* sample article, there is only a single theorem.

In the *secondarticle.tex* sample article, there are a number of different proclamations in a variety of styles, with varying degrees of emphasis.

The two steps are required to use a proclamation:

Step 1:

Define the proclamation with a \newtheorem command *in the preamble* of the document. For instance, the line

\newtheorem{theorem}{Theorem}

defines a theorem environment.

Step 2:

Invoke the proclamation as an environment *in the body* of our document. Using the proclamation definition from Step 1, type

```
\begin{theorem}
   My first theorem.
\end{theorem}
```

to produce a theorem:

Theorem 1. *My first theorem.*

In the proclamation definition

\newtheorem{theorem}{Theorem}

the first argument, *theorem*, is the name of the environment that invokes the theorem. The second argument, *Theorem*, is the name that is used when the proclamation is typeset. LATEX numbers the theorems automatically and typesets them with vertical space above and below. The phrase **Theorem 1.** appears, followed by the theorem itself, which may be emphasized. Of course, the formatting of the theorem depends on the document class and the proclamation style.

We may also specify an optional argument,

```
\begin{theorem}[The Fuchs-Schmidt Theorem]
   The statement of the theorem.
\end{theorem}
```

that appears as the name of the theorem:

Theorem 1 (The Fuchs-Schmidt Theorem). *The statement of the theorem.*

Consecutive numbering:

If we want to number two sets of proclamations consecutively, we can do so by first defining one proclamation, and then using its name as an optional argument of the second proclamation. For example, to number the lemmas and propositions in our paper consecutively, we type the following two lines in your preamble:

\newtheorem{lemma}{Lemma}
\newtheorem{proposition}[lemma]{Proposition}

Lemmas and propositions are then consecutively numbered as **Lemma 1, Proposition 2, Proposition 3**, and so on.

Numbering within a section:

The \newtheorem command may also have a different optional argument; it causes LATEX to number the proclamations within sections. For example,

\newtheorem{lemma}{Lemma}[section]

numbers the lemmas in Section 1 as **Lemma 1.1** and **Lemma 1.2**. In Section 2, you have **Lemma 2.1** and **Lemma 2.2**, and so on.

Instead of section, we may use any sectioning command provided by the document class, such as chapter, section, and subsection.

Consecutive numbering and numbering within a section can be combined. For example,

\newtheorem{lemma}{Lemma}[section]

\newtheorem{proposition}[lemma]{Proposition}

sets up the lemma and proposition environments so that they are numbered consecutively within sections: **Lemma 1.1, Proposition 1.2, Proposition 1.3** and **Proposition 2.1, Lemma 2.2**, and so on.

2.4.1 The full syntax:

The full form of \newtheorem is

\newtheorem{envname }[procCounter]{Name }[secCounter]

where the two optional arguments are mutually exclusive, and ***envname*** is the name of the environment to be used in the body of the document. For instance, we may use theorem for the *envname* of a theorem, so that a theorem is typed inside a theorem environment. Of course, *envname* is just a label; we are free to choose any environment name, such as thm or George (as long as *the name is not in use as the name of another command or environment*). This argument is also the name of the counter LATEX uses to number these text environments.

procCounter is an optional argument. It sets the new proclamation to use the counter of a previously defined proclamation and the two proclamations are consecutively numbered.

Name is the text that is typeset when the proclamation is invoked. So, if Theorem is given as *Name*, then we get **Theorem 1, Theorem 2**, and so on in our document.

secCounter is an optional argument that causes the *Name* environments to be numbered within the appropriate sectioning units. So if theorem is the *envname* and section is the *secCounter*, then in Section 1 we have **Theorem 1.1, Theorem 1.2,** and so on. In Section 2 we get **Theorem 2.1, Theorem 2.2,** and so on. Proclamations may be

numbered within subsections, sections, chapters, or any other sectioning unit automatically numbered by LATEX.

2.4.2 Proclamations with style:

We can choose one of three styles for our proclamations by preceding the definitions with the \theoremstyle{style} command, where *style* is one of the following:

- plain, the most emphatic
- definition
- remark, the least emphatic

There are a few extra options, including the \newtheorem* command, an unnumbered version of \newtheorem.

The following commands set the styles in the *secondarticle.tex* article. The typeset sample article shows how the chosen styles affect the typeset proclamations.

```
\theoremstyle{plain}
\newtheorem{theorem}{Theorem}
\newtheorem{corollary}{Corollary}
\newtheorem*{main}{Main Theorem}
\newtheorem{lemma}{Lemma}
\newtheorem{proposition}{Proposition}

\theoremstyle{definition}
\newtheorem{definition}{Definition}

\theoremstyle{remark}
\newtheorem*{notation}{Notation}
```

A proclamation created by a *\newtheorem* command has the style of the last *\theoremstyle* command preceding it. The default style is plain.

2.5 Proof environments:

A proof is the contents of a proof environment. For instance,

Proof. This is a proof, delimited by the q.e.d. symbol.

typed as

```
\begin{proof}
This is a proof, delimited by the q.e.d.\ symbol.
\end{proof}
```

A proof is set off from the surrounding text with some vertical space. The end of the proof is marked with the symbol at the end of the line. There are a few examples of the proof environment in the *secondarticle.tex* sample article.

Lists in proofs:

If a proof starts with a list environment, precede the list by \hfill.
If we want to suppress the symbol at the end of a proof, give the command

```
\begin{proof}
   ...
   \renewcommand{\qedsymbol}{}
\end{proof}
```

To suppress the end of the proof symbol in the whole article, give the

```
\renewcommand{\qedsymbol}{}
```

command in the preamble.

To substitute another phrase for *Proof*, such as *Necessity*, as in

Necessity. This is the proof of necessity.

use the proof environment with an optional argument:

```
\begin{proof}[Necessity]
This is the proof of necessity.
\end{proof}
```

The optional argument may contain a reference, as in

```
\begin{proof}[Proof of Theorem~\ref{T:smooth}]
```

which might be typeset as

Proof of Theorem 5. This is the proof. □

It is easy to make the mistake of placing the optional argument after `\begin`:

```
\begin[Proof of Theorem~\ref{T:P*}]{proof}
```

You get a message

```
! LaTeX Error: Bad math environment delimiter.

l.91 \begin{equation}
                      \label{E:cong2}
```

which is not very helpful.

There is a problem with the placement of the q.e.d. symbol if the proof ends with a displayed formula (or a list environment). For instance,

```
\begin{proof}
Now the proof follows from the equation
\[
   a^2 = b^2 + c^2.
\]
\end{proof}
```

typesets as

Proof. Now the proof follows from the equation

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

□

To correct the placement of the q.e.d. symbol, use the \qedhere command:

```
\begin{proof}
Now the proof follows from the equation
\[
   a^2 = b^2 + c^2.\qedhere
\]
\end{proof}
```

which typesets as

Proof. Now the proof follows from the equation

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

□

2.6 Tabular environments:

A tabular environment creates a table that LATEX treats as a "large symbol". In particular, a table cannot be broken across pages.

Here is a simple table,

Name	1	2	3
Peter	2.45	34.12	1.00
John	0.00	12.89	3.71
David	2.00	1.85	0.71

, typeset inline. This looks awful, but it does make the point that the table is just a "large symbol". The table is typed as

```
\begin{tabular}{ | l | r | r | r | }
   \hline
   Name     & 1     & 2     & 3    \\ \hline
   Peter    &  2.45 & 34.12 & 1.00\\ \hline
   John     &  0.00 & 12.89 & 3.71\\ \hline
   David    &  2.00 & 1.85  & 0.71\\ \hline
\end{tabular}
```

Name	1	2	3
Peter	2.45	34.12	1.00
John	0.00	12.89	3.71
David	2.00	1.85	0.71

Table 2.1: Tabular table.

```
\begin{table}
   \begin{center}
     \begin{tabular}{ | l | r | r | r | }
       \hline
       Name     & 1     & 2     & 3   \\ \hline
       Peter    &  2.45 & 34.12 & 1.00\\ \hline
       John     &  0.00 & 12.89 & 3.71\\ \hline
       David    &  2.00 & 1.85  & 0.71\\ \hline
     \end{tabular}
     \caption{Tabular table.}\label{Ta:first}
   \end{center}
\end{table}
```

This table is displayed as Table 2.1.

Tabular **Environments:**

1. \begin{tabular} requires an argument consisting of a character l, r, or c, meaning left, right, or center alignment, for each column, and optionally, the | symbols. Each | indicates a vertical line in the typeset table. Spaces in the argument are ignored but can be used for readability.

2. Columns are separated by ampersands (&) and rows are separated by \\.

3. & absorbs spaces on either side.

4. The \hline command creates a horizontal rule in the typeset table. It is placed either at the beginning of the table (after the \begin line) or it must follow a \\ command.

5. If we use a horizontal line to finish the table, we must separate the last row of the table from the \hline command with the \\ command.

6. \begin{tabular} takes an optional argument, b or t, to specify the bottom or the top vertical alignment of the table with the baseline. The default is center alignment.

The example of table is given by

```
\begin{tabular}{ | p{1in} | r | r | r | }\hline
   Name     &  1    & 2     & 3   \\ \hline
   Peter    &  2.45 & 34.12 & 1.00\\ \hline
   John     &  0.00 & 12.89 & 3.71\\ \hline
   David    &  2.00 & 1.85  & 0.71\\ \hline
\end{tabular}
```

which typesets as

Name	1	2	3
Peter	2.45	34.12	1.00
John	0.00	12.89	3.71
David	2.00	1.85	0.71

2.7 Tabbing environments:

Although of limited use for mathematical typesetting, the tabbing environment can be useful for typing algorithms, computer programs, and so forth. LATEX calculates the width of a column in the tabular environment based on the widest entry. The tabbing environment allows us to control the width of the columns. The \\ command is the line separator, tab stops are set by \= and are remembered by LATEX in the order they are given, and \> moves to the next tab position. We can easily reset tab positions. For instance, if we are past the second tab position by using \> twice, and there is a third tab position, the \= command resets it. Lines of comments may be inserted with the \kill command, see the examples below, or with the % character. The difference is that a line with \kill can be used to set tab stops, whereas a commented-out line cannot.

A simple example:

```
PrintTime
     Block[timing],
          timing = Timing[expr];
          Print[ timing[[1]] ];
     ]
End[]
```

typed as

```
{\ttfamily
\begin{tabbing}
   Print\=Time\\
   \>Block\=[timing,\\
   \>\>timing = Timing[expr];\\
   (careful with initialization)\kill
   \>\>Print[ timing[[1]] ];\\
   \>]\\
   End[]
\end{tabbing}
}% end \ttfamily
```

An alternative method is to use a line to set the tab stops, and then `\kill` the line so it does not print:

```
{\ttfamily
\begin{tabbing}
   \hspace*{.25in}\=\hspace{2ex}\=\hspace{2ex}\=
```

```
      \hspace{2ex}\kill
  \>  $k := 1$\\
  \>  $l_k := 0$; $r_k := 1$\\
  \>  loop\\
  \>  \> $m_k := (l_k + r_k)/2$\\
  \>  \> if $w < m_k$ then\\
  \>  \>  \> $b_k := 0$; $r_k := m_k$\\
  \>  \> else if $w > m_k$ then\\
  \>  \>  \> $b_k := 1$; $l_k := m_k$\\
  \>  \> end if\\
  \>  \> $k := k + 1$\\
  \> end loop
\end{tabbing}
}% end \ttfamily
```

which typesets as

```
k := 1
l_k := 0; r_k := 1
loop
  m_k := (l_k + r_k)/2
  if w < m_k then
    b_k := 0; r_k := m_k
  else if w > m_k then
    b_k := 1; l_k := m_k
  end if
  k := k + 1
end loop
```

Some simple rules:

- There is no \\ command on a line containing the \kill command.
- We may set the tabs in a \kill line with \hspace commands.
- The \> command moves to the next tab stop, even if the text we have already typed extends past that stop, which can result in overprinting.
- The tabbing environment has to be typeset with typewriter style font—note the \ttfamily command.

To illustrate the rule, type

```
\begin{tabbing}
   This is short.\=\\
   This is much longer, \> and jumps back.
\end{tabbing}
```

which typesets as

This is short.
This is much longer, and jumps back.

2.8 Miscellaneous displayed text environments:

There are four more displayed text environments, of limited use in math: quote, quotation, verse, and verbatim.

We also discuss an inline version of the verbatim environment, the \verb command.

Quotes:

The quote environment is used for short (one paragraph) quotations:

It's not that I'm afraid to die. I just don't want to be there when it happens. *Woody Allen*

Literature is news that STAYS news. *Ezra Pound*

which is typed as:

```
\begin{quote}
   It's not that I'm afraid to die. I just don't
   want to be there when it happens.
   \emph{Woody Allen}

   Literature is news that STAYS news.
   \emph{Ezra Pound}
\end{quote}
```

Note that multiple quotes are separated by blank lines.

Quotations:

In the quotation environment, blank lines mark new paragraphs:

KATH: Can he be present at the birth of his child?

ED: It's all any reasonable child can expect if the dad is present at the conception.

Joe Orton

is typed as

```
\begin{quotation}
   KATH: Can he be present at the birth of his child?

   ED: It's all any reasonable child can expect
   if the dad is present at the conception.
   \begin{flushright}
      \emph{Joe Orton}
   \end{flushright}
\end{quotation}
```

Verses:

A verse environment,

I think that I shall never see
A poem lovely as a tree.

Poems are made by fools like me,
But only God can make a tree.

Joyce Kilmer

is typed as

```
\begin{verse}
   I think that I shall never see\\
   A poem lovely as a tree.

   Poems are made by fools like me,\\
   But only God can make a tree.

   \begin{flushright}
      \emph{Joyce Kilmer}


    \end{flushright}
 \end{verse}
```

Lines are separated by \\ and stanzas by blank lines. Long lines are typeset with hanging indent.

Verbatim typesetting:

Finally, there is the verbatim text environment.

Formula (2) in Section 3 should be typed as follows:

```
\begin{equation}
D = \{ x_0 \mid x_0 \Rightarrow a_1 \} \tag{2}
\end{equation}
```

Please make the necessary corrections.

The problem is that if we just type

```
Formula (2) in Section 3 should be typed as follows:
\begin{equation}
    D = \{ x_0 \mid x_0 \Rightarrow a_1 \} \tag{2}
\end{equation}
Please make the necessary corrections.
```

it typesets as

Formula (2) in Section 3 should be typed as follows:

$$D = \{x_0 \mid x_0 \Rightarrow a_1\} \tag{2}$$

Please make the necessary corrections.

To get the proper typeset form, type it as follows:

```
Formula (2) in Section 3 should be typed as follows:
\begin{verbatim}
\begin{equation}
D = \{ x_0 \mid x_0 \Rightarrow a_1 \} \tag{2}
\end{equation}
\end{verbatim}
Please make the necessary corrections.
```

Unit-III:

Typing math: Math environments - spacing rules - equations - spacing rules - equations - Basic constructs - Arithmetic operations - Delimiters – Operators - Math accents - Stretchable horizontal lines - formula gallery.

3.0 Typing math:

Introduction:

LATEX was designed for typesetting math. A math formula can be typeset *inline*, as part of the current paragraph, or *displayed*, on a separate line or lines with vertical space before and after the formula. In this and the next chapter we discuss formulas that are set inline or displayed on a *single line.*

We start with a discussion of LATEX's basic math environments, spacing rules in math, and continue with the equation environment. The basic constructs of a formula—arithmetic (including subscripts and superscripts), binomial coefficients, ellipses, integrals, roots, and text—are discussed in detail. From the basic constructs of that section, we can build very complicated formulas, one step at a time.

Also, we discuss delimiters, operators, and math accents. Then, we discuss three types of stretchable horizontal lines that can be used above or below a formula: braces, bars, and arrows. There are also stretchable arrow math symbols and *Formula Gallery.*

3.1 Math environments:

A formula in a LATEX document can be typeset *inline,* like the congruence $a \equiv b\ (\theta)$ or the integral $\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$, or *displayed*, as in

$$a \equiv b \quad (\theta)$$

or

$$\int_{-\infty}^{\infty} e^{-x^2}\, dx = \sqrt{\pi}$$

Notice how changing these two formulas from inline to displayed affects their appearance.

Inline and displayed math formulas are typeset using the *math environments* math and *displaymath*, respectively. Because math formulas occur so frequently, LATEX has abbreviations: the special braces \(and \) or $ are used for the math environment, and \[and \] for the *displaymath* environment. So, our inline example may be typed as

```
$a \equiv b \pod{\theta}$
```

or

```
\( a \equiv b \pod{\theta} \)
```

or

```
\begin{math}
   a \equiv b \pod{\theta}
\end{math}
```

The displayed example can be typed as

```
\[
  \int_{-\infty}^{\infty} e^{-x^{2}} \, dx = \sqrt{\pi}
\]
```

or

```
\begin{displaymath}
   \int_{-\infty}^{\infty} e^{-x^{2}} \, dx = \sqrt{\pi}
\end{displaymath}
```

No blank lines are permitted in a math or *displaymath* environment.

3.2 Spacing rules:

In text, the most important spacing rule is that any number of spaces in the source file equals one space in the typeset document. The spacing rule for math mode is even more straightforward.

LATEX ignores spaces in math.

In other words, all spacing in math mode is provided by LATEX. For instance,

$a+b=c$

and

$a + b = c$

are both typeset as $a + b = c$.

There are two exceptions to this rule:

1. A space indicating the end of a command name is recognized. For instance, in
   $a \quad b$
   LATEX does not ignore the space between \quad and b.
2. If we switch back to text mode inside a math formula with a \text command, then the text spacing rules apply in the argument of such a command.

LATEX provides controls for spaces in typeset math. The spaces we type in math do not affect the typeset document.

3.3 Equations:

An *equation* is a numbered formula displayed on a single typeset line.

Equations are typed in an equation environment. The equation environment and *displaymath* environment are exactly the same except that the equation environment assigns a number to the displayed formula

(1) $\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$

This example is typed as

```
\begin{equation}\label{E:int}
   \int_{-\infty}^{\infty} e^{-x^{2}} \, dx = \sqrt{\pi}
\end{equation}
```

The \label command in the equation environment is optional. If we use a \label command, the number assigned to the equation can be referenced with the \ref command. So

see~(\ref{E:int})

typesets as see (1). Even better, use the \eqref command, which places the parentheses automatically:

see~\eqref{E:int}

also typesets as see (1). In fact, the \eqref command does more: It typesets the reference *upright,* even in italicized or slanted text.

Analogously, the \upn command forces the use of upright characters for digits, punctuations, parentheses, etc. LATEX numbers equations consecutively. As a rule, equations are numbered consecutively throughout articles, whereas in books, numbering starts from 1 at the start of each chapter. We may also choose to have equations numbered within each section—(1.1), (1.2), . . . , in Section 1; (2.1), (2.2), . . . , in Section 2; and so on—by including the command

\numberwithin{equation}{section}

in the preamble of our document.

Equation environment:

1. No blank lines are permitted within an equation or equation∗ environment.

2. No blank line before the environment.

3.4 Basic constructs:

A formula is built by combining various basic constructs. This section discusses the following constructs:

- Arithmetic operations – Subscripts and superscripts
- Binomial coefficients
- Ellipses
- Integrals
- Roots
- Text
- Hebrew and Greek letters

3.5 Arithmetic operations:

The *arithmetic operations* are typed pretty much as we would expect. To get $a + b,\ a - b,\ -a, a/b$, and ab, type

$a + b$, $a - b$, $-a$, a / b, $a\,b$

There are two other forms of multiplication and one of division: $a \cdot b, a \times b$, and $a \div b$. They are typed as follows:

$a \cdot b$, $a \times b$, $a \div b$

In displayed formulas, *fractions* are usually typed with the *\frac* command. To get

$$\frac{1 + 2x}{x + y + xy}$$

type

```
\[
   \frac{1 + 2x}{x + y + xy}
\]
```

We can use display-style fractions inline with *\dfrac*, and inline-style fractions in displayed math environments with *\tfrac*; for example, $\frac{3+a^2}{4+b}$ is typed as

```
$\dfrac{3 + a^{2}}{4 + b}$
```

and

$\frac{3+a^2}{4+b}$

is typed as

```
\[
  \tfrac{3 + a^{2}}{4 + b}
\]
```

The \dfrac command is often used in matrices whose entries would look too small with the \frac command.

Subscripts and superscripts:

Subscripts are typed with _ and *superscripts* with ^. Remember to enclose the subscripted or superscripted expression in braces:

```
\[
   a_{1},\ a_{i_{1}},\ a^{2},\  a^{b^{c}},\ a^{i_{1}},\
      a_{i} + 1,\ a_{i + 1},\ a_{1}^{2},\ a^{2}_{1}
\]
```

typesets as

$$a_1,\ a_{i_1},\ a^2,\ a^{b^c},\ a^{i_1},\ a_i + 1,\ a_{i+1},\ a_1^2,\ a_1^2$$

For a^{b^c}, type `$a^{b^{c}}$`, not `$a^{b}^{c}$`. If you type the latter, you get the message

```
! Double superscript.
```

Similarly, a_{b_c} is typed as `$a_{b_{c}}$`, not as `$a_{b}_{c}$`.

In many instances, the braces for the subscripts and superscripts could be omitted, but we should type them anyway.

3.5.1 Binomial coefficients:

Binomials are typeset with the *\binom* command. Here are two examples shown inline,

$\binom{a}{b+c}$ and $\binom{\frac{n^2-1}{2}}{n+1}$, and displayed:

$$\binom{a}{b+c} \text{ and } \binom{\frac{n^2-1}{2}}{n+1}$$

The latter is typed as

```
\[
   \binom{a}{b + c} \text{ and }
   \binom{\frac{n^{2} - 1}{2}}{n + 1}
\]
```

We can use display-style binomials inline with \dbinom, and inline-style binomials in displayed math environments with \tbinom. For example, $\binom{a}{b+c}$ is typed as *$\dbinom{a}{b + c}$*.

3.5.2 Ellipses:

There are two types of *ellipsis* in math, the *low* or *on-the-line ellipsis*, as in

$$F(x_1, x_2, \ldots, x_n)$$

and the *centered ellipsis*, as in

$$x_1 + x_2 + \cdots + x_n$$

These two formulas are typed as

```
\[
   F(x_{1}, x_{2}, \dots, x_{n})
\]
```

and

```
\[
   x_{1} + x_{2} + \dots + x_{n}
\]
```

LATEX uses the symbol following a \dots command to decide whether to use a low or centered ellipsis. If it fails to make the right decision as in

$$\alpha(x_1 + x_2 + \ldots)$$

typed as

```
\[
   \alpha(x_{1} + x_{2} + \dots)
\]
```

help LaTeX by giving the command `\ldots` for low and `\cdots` for centered ellipsis. So to get the last formula right, type

```
\[
   \alpha(x_{1} + x_{2} + \cdots)
\]
```

and it typesets correctly:

$$\alpha(x_1 + x_2 + \cdots)$$

There are five more variants of the \dots command:

- \dotsc for an ellipsis followed by a comma
- \dotsb for an ellipsis followed by a binary operation or relation
- \dotsm for an ellipsis followed by multiplication
- \dotsi for an ellipsis with integrals
- \dotso for an “other” ellipsis

These commands not only force the ellipsis to be low or centered, but also adjust the spacing.

3.5.3 Integrals:

We have already seen the formula $\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$ in both inline and displayed forms in the first section of this chapter. The lower limit is typeset as a subscript and the upper limit is typeset as a superscript. To force the limits below and above the integral symbol, use the \limits command. The \nolimits command does the reverse.

To typeset $\int\limits_{-\infty}^{\infty} e^{-x^2}\, dx = \sqrt{\pi}$, type

```
$\int\limits_{-\infty}^{\infty} e^{-x^{2}} \, dx = \sqrt{\pi}$
```

There are five commands to produce variants of the basic integral symbol:

```
\oint  \iint  \iiint  \iiiint  \idotsint
```

which typeset as

$\oint \quad \iint \quad \iiint \quad \iiiint \quad \int \cdots \int$

3.5.4 Roots:

The \sqrt command produces a square root, for instance,

`$\sqrt{5}$`	typesets as	$\sqrt{5}$
`$\sqrt{a + 2b + c^{2}}$`	typesets as	$\sqrt{a + 2b + c^2}$

Here is a more interesting example:

$$\sqrt{1+\sqrt{1+\frac{1}{2}\sqrt{1+\frac{1}{3}\sqrt{1+\frac{1}{4}\sqrt{1+\cdots}}}}}$$

typed as

```
\[
   \sqrt{1 + \sqrt{1 + \frac{1}{2}\sqrt{1 + \frac{1}{3}
    \sqrt{1 + \frac{1}{4}\sqrt{1 + \cdots}}}}}
\]
```

For n-th roots other than the square root, that is, $n \neq 2$, specify n with an optional argument. To get $\sqrt[3]{5}$, type `$\sqrt[3]{5}$`.

3.5.5 Text in math:

LATEX allows we to include text in formulas with the \text command. The formula

$$A = \{x \mid x \in X_i, \text{ for some } i \in I\}$$

is typed as

```
\[
   A = \{ x \mid x \in X_{i}, \text{ for some }
       i \in I \}
\]
```

Note that we have to leave space before for and after some inside the argument of \text. The argument of the \text command is always typeset in a single line.

Sometimes it is more convenient to go into math mode within the argument of a \text command rather than end the \text and start another, as in

$$A = \{x \mid \text{for } x \text{ large}\}$$

which may be typed as

```
\[
   A = \{ x \mid \text{for $x$ large} \}
\]
```

The \text command correctly sizes its argument to match the context. The formula

$$a_{\text{left}} + 2 = a_{\text{right}}$$

is typed as

```
\[
   a_{\text{left}} + 2 = a_{\text{right}}
\]
```

Note that \text typesets its argument *in the size and shape* of the surrounding text. If we want the text in a formula to be typeset in the document font family independent of the surrounding text, use

\textnormal{ ... }

or

{\normalfont ...}

For instance, if you have a constant a_{right}, then in a theorem:

Theorem 1. *The constant a_{right} is recursive in a.*

The subscript is wrong. To get it right, type the constant as

```
$a_{\normalfont\text{right}}$
```

Now the theorem typesets as

Theorem 1. *The constant a_{right} is recursive in a.*

For instance, \textbf uses the size and shape of the surrounding text to typeset its argument in bold (extended).

3.5.6 Hebrew and Greek letters:

Math uses only four Hebrew letters: $\aleph$, $\beth$, $\daleth$, $\gimel$, typed as

\com{aleph}, \com{beth}, \com{daleth}, \com{gimel}

The 26 Greek letters come in lower case and some also in upper case. There is no upper case α, because it is the same as the Latin letter A. Seven lower case Greek letter also come in a variant. For instance, the variant of ϕ is φ. Tables 3.1 and 3.2 list them all.

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

Table 3.1: Lowercase Greek letters

Type	Typeset	Type	Typeset	Type	Typeset
\Gamma	Γ	\Xi	Ξ	\Phi	Φ
\Delta	Δ	\Pi	Π	\Psi	Ψ
\Theta	Θ	\Sigma	Σ	\Omega	Ω
\Lambda	Λ	\Upsilon	Υ		
\varGamma	$\varGamma$	\varXi	$\varXi$	\varPhi	$\varPhi$
\varDelta	$\varDelta$	\varPi	$\varPi$	\varPsi	$\varPsi$
\varTheta	$\varTheta$	\varSigma	$\varSigma$	\varOmega	$\varOmega$
\varLambda	$\varLambda$	\varUpsilon	$\varUpsilon$		

Table 3.2: Uppercase Greek letters

3.6 Delimiters:

Delimiters are used to enclose some subformulas. In the following formula we use two delimiters: parentheses and square brackets: $[(a * b) + (c * d)]^2$; this typesets as $[(a * b) + (c * d)]^2$. LATEX knows that parentheses and square brackets are delimiters, and spaces them accordingly.

The standard delimiters are shown in Table 3.3. Note that delimiters are math symbols with special spacing rules and we can use them in any way we please, not only in pairs. LATEX does not stop us from typing \uparrow(x], which typesets as ↑ (x].

Observe the difference in spacing between $|| a ||$ and $\|a\|$. The first, $|| a ||$, was typed incorrectly as $|| a ||$. As a result, the vertical bars are too far apart. The second was typed correctly using the appropriate delimiter commands: $\| a \|$. Here they are again side-by-side, enlarged:

$$|| a || \quad \|a\|$$

Name	Type	Typeset
left parenthesis	(	(
right parenthesis	)	)
left bracket	[or \lbrack	[
right bracket	] or \rbrack	]
left brace	\{ or \lbrace	{
right brace	\} or \rbrace	}
backslash	\backslash	\
forward slash	/	/
left angle bracket	\langle	⟨
right angle bracket	\rangle	⟩
vertical line	| or \vert	|
double vertical line	\\| or \Vert	‖
left floor	\lfloor	⌊
right floor	\rfloor	⌋
left ceiling	\lceil	⌈
right ceiling	\rceil	⌉
upward	\uparrow	↑
double upward	\Uparrow	⇑
downward	\downarrow	↓
double downward	\Downarrow	⇓
up-and-down	\updownarrow	↕
double up-and-down	\Updownarrow	⇕
upper-left corner	\ulcorner	⌜
upper-right corner	\urcorner	⌝
lower-left corner	\llcorner	⌞
lower-right corner	\lrcorner	⌟

Table 3.3: Standard delimiters.

3.7 Operators:

We cannot just type sin x to typeset the sine function in math mode. Indeed,

$$\$sin\ x\$$$

produces $sinx$ instead of $sin\ x$, as we intended. Type this function as

$$\$\backslash sin\ x\$$$

The $\backslash sin$ command prints sin with the proper style and spacing. LATEX calls $\backslash sin$ an

operator or log-*like function*.

3.7.1 Operator tables:

There are two types of operators:

1. *Operators without limits*, such as $\sin$
2. *Operators with limits*, such as $\lim$, that take a subscript in inlinemode and a "limit" in displayed math mode. For example, $\log_{x\to 0} f(x) = 1$ is typed as

   `$\lim_{x \to 0} f(x) = 1$`

The same formula displayed,

$$\lim_{x\to 0} f(x) = 1$$

is typed as

```
\[
   \lim_{x \to 0} f(x) = 1
\]
```

The operators are listed in Tables 3.4 and 3.5. The entries in the last two rows of Table 3.5 can be illustrated by

$$\varinjlim_{x\to 0} \quad \varprojlim_{x\to 0} \quad \varliminf_{x\to 0} \quad \varlimsup_{x\to 0}$$

Type	Typeset	Type	Typeset	Type	Typeset	Type	Typeset
\arccos	arccos	\cot	cot	\hom	hom	\sin	sin
\arcsin	arcsin	\coth	coth	\ker	ker	\sinh	sinh
\arctan	arctan	\csc	csc	\lg	lg	\tan	tan
\arg	arg	\deg	deg	\ln	ln	\tanh	tanh
\cos	cos	\dim	dim	\log	log		
\cosh	cosh	\exp	exp	\sec	sec		

Table 3.4: Operators without limits.

which are typed as

```
\[
   \varliminf_{x \to 0} \quad \varlimsup_{x \to 0} \quad
   \varinjlim_{x \to 0} \quad \varprojlim_{x \to 0}
\]
```

Type	Typeset	Type	Typeset
\det	det	\limsup	lim sup
\gcd	gcd	\max	max
\inf	inf	\min	min
\lim	lim	\Pr	Pr
\liminf	lim inf	\sup	sup
\injlim	inj lim	\projlim	proj lim
\varliminf	$\underline{\lim}$	\varlimsup	$\overline{\lim}$
\varinjlim	$\varinjlim$	\varprojlim	$\varprojlim$

Table 3.5: Operators with limits.

$$\operatorname{inj\,lim}_{x\to 0} \quad \operatorname{lim\,inf}_{x\to 0} \quad \operatorname{lim\,sup}_{x\to 0} \quad \operatorname{proj\,lim}_{x\to 0}$$

These operators were typed as

```
\[
   \injlim_{x \to 0} \quad \liminf_{x \to 0} \quad
   \limsup_{x \to 0} \quad \projlim_{x \to 0}
\]
```

For example, the formulas

$$\operatorname{inj\,lim}_{x\to 0} \quad \operatorname{lim\,inf}_{x\to 0} \quad \operatorname{lim\,sup}_{x\to 0} \quad \operatorname{proj\,lim}_{x\to 0}$$

are typed as

```
\[
   \injlim\nolimits_{x \to 0} \quad
   \liminf\nolimits_{x \to 0} \quad
   \limsup\nolimits_{x \to 0} \quad
   \projlim\nolimits_{x \to 0}
\]
```

3.7.2 Congruences:

\mod is a special operator used for congruences. Congruences are usually typeset using the \pmod or \pod variant. There is also the \bmod command, which is used as a binary operation. All four commands are shown in Table 3.6.

Type	Typeset
`$a \equiv v \mod{\theta}$`	$a \equiv v \mod \theta$
`$a \bmod b$`	$a \bmod b$
`$a \equiv v \pmod{\theta}$`	$a \equiv v \pmod \theta$
`$a \equiv v \pod{\theta}$`	$a \equiv v \ (\theta)$

Table 3.6: Congruences.

3.7.3 Large operators:

Here is a sum typeset inline, $\sum_{i=1}^{n} x_i^2$, and displayed,

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

In the latter form, the sum symbol is larger. Operators that behave in this way are called *large operators*. Table 3.7 gives a complete list of large operators.

We can use the \nolimits command if we wish to show the limits of large operators as subscripts and superscripts in a displayed math environment.

The formula

$$\bigsqcup_{\mathfrak{m}} X = a$$

is typed as

```
\[
   \bigsqcup\nolimits_{ \mathfrak{m} } X = a
\]
```

Sums and products are very important constructs. The examples

$$\frac{z^d - z_0^d}{z - z_0} = \sum_{k=1}^{d} z_0^{k-1} z^{d-k} \quad \text{and} \quad (T^n)'(x_0) = \prod_{k=0}^{n-1} T'(x_k)$$

are typed as

```
\[
   \frac{z^{d} - z_{0}^{d}}
        {z - z_{0}} =
   \sum_{k = 1}^{d} z_{0}^{k - 1} z^{d - k}
   \text{\quad and\quad}
   (T^{n})'(x_{0}) = \prod_{k=0}^{n - 1} T'(x_{k})
\]
```

Type	Inline	Displayed
`\int_{a}^{b}`	$\int_a^b$	$\displaystyle\int_a^b$
`\oint_{a}^{b}`	$\oint_a^b$	$\displaystyle\oint_a^b$
`\iint_{a}^{b}`	$\iint_a^b$	$\displaystyle\iint_a^b$
`\iiint_{a}^{b}`	$\iiint_a^b$	$\displaystyle\iiint_a^b$
`\iiiint_{a}^{b}`	$\iiiint_a^b$	$\displaystyle\iiiint_a^b$
`\idotsint_{a}^{b}`	$\int\cdots\int_a^b$	$\displaystyle\int\cdots\int_a^b$
`\prod_{i=1}^{n}`	$\prod_{i=1}^{n}$	$\displaystyle\prod_{i=1}^{n}$
`\coprod_{i=1}^{n}`	$\coprod_{i=1}^{n}$	$\displaystyle\coprod_{i=1}^{n}$
`\bigcap_{i=1}^{n}`	$\bigcap_{i=1}^{n}$	$\displaystyle\bigcap_{i=1}^{n}$
`\bigcup_{i=1}^{n}`	$\bigcup_{i=1}^{n}$	$\displaystyle\bigcup_{i=1}^{n}$
`\bigwedge_{i=1}^{n}`	$\bigwedge_{i=1}^{n}$	$\displaystyle\bigwedge_{i=1}^{n}$
`\bigvee_{i=1}^{n}`	$\bigvee_{i=1}^{n}$	$\displaystyle\bigvee_{i=1}^{n}$
`\bigsqcup_{i=1}^{n}`	$\bigsqcup_{i=1}^{n}$	$\displaystyle\bigsqcup_{i=1}^{n}$
`\biguplus_{i=1}^{n}`	$\biguplus_{i=1}^{n}$	$\displaystyle\biguplus_{i=1}^{n}$
`\bigotimes_{i=1}^{n}`	$\bigotimes_{i=1}^{n}$	$\displaystyle\bigotimes_{i=1}^{n}$
`\bigoplus_{i=1}^{n}`	$\bigoplus_{i=1}^{n}$	$\displaystyle\bigoplus_{i=1}^{n}$
`\bigodot_{i=1}^{n}`	$\bigodot_{i=1}^{n}$	$\displaystyle\bigodot_{i=1}^{n}$
`\sum_{i=1}^{n}`	$\sum_{i=1}^{n}$	$\displaystyle\sum_{i=1}^{n}$

Table 3.7: Large operators.

3.7.4 Multiline subscripts and superscripts

The \substack command provides multiline limits for large operators. For instance,

$$\sum_{\substack{i<n\\ i \text{ even}}} x_i^2$$

is typed as

```
\[
   \sum_{ \substack{ i < n\\
                     i \text{ even} } }
    x_{i}^{2}
\]
```

There is only one rule to remember. Use the line separator command \\. We can use the \substack command wherever subscripts or superscripts are used.

3.8 Math accents:

The accents used in text cannot be used in math formulas. For accents in formulas a separate set of commands is provided. All math accents are shown in Table 3.8. The *amsxtra* package is needed for the accents in the second column. To use them, make sure to place in the preamble the line

\usepackage{amsxtra}

We can also use double accents, such as

```
\[
   \hat{\hat{A}}
\]
```

which typesets as $\hat{\hat{A}}$.

The two "wide" varieties, \widehat and \widetilde, expand to fit the symbols (their arguments) covered: $\widehat{A}$, $\widehat{ab}, \widehat{ıı}, \widehat{aıa}ı, \widehat{ıııı}$ and $\widetilde{A}, \widetilde{ab}, \widetilde{ıı}, \widetilde{aıa}ı, \widetilde{ıııı}$ (the last example is typed as $\widetilde{iiiii}$). If the base is too wide, the accent is centered:

$$\widehat{ABCDE}$$

		amsxtra	
Type	Typeset	Type	Typeset
\acute{a}	$\acute{a}$		
\bar{a}	$\bar{a}$		
\breve{a}	$\breve{a}$	\spbreve	$\spbreve$
\check{a}	$\check{a}$	\spcheck	$\spcheck$
\dot{a}	$\dot{a}$	\spdot	$\spdot$
\ddot{a}	$\ddot{a}$	\spddot	$\spddot$
\dddot{a}	$\dddot{a}$	\spdddot	$\spdddot$
\ddddot{a}	$\ddddot{a}$		
\grave{a}	$\grave{a}$		
\hat{a}	$\hat{a}$		
\widehat{a}	$\widehat{a}$	\sphat	$\sphat$
\mathring{a}	$\mathring{a}$		
\tilde{a}	$\tilde{a}$		
\widetilde{a}	$\widetilde{a}$	\sptilde	$\sptilde$
\vec{a}	$\vec{a}$		

Table 3.8: Math accents

3.9 Stretchable horizontal lines:

LATEX provides three types of stretchable horizontal lines that appear above or below a formula, braces, bars, and arrows. There are also stretchable arrow math symbols.

3.9.1 Horizontal braces:

The \overbrace command places a brace of variable size above its argument, as in

$$\overbrace{a + b + \cdots + z}$$

which is typed as

```
\[
   \overbrace{a + b + \dots + z}
\]
```

A superscript adds a label to the brace, as in

$$\overbrace{a + a + \cdots + a}^{n}$$

which is typed as

```
\[
   \overbrace{a + a + \dots + a}^{n}
\]
```

The \underbrace command works similarly, placing a brace below its argument. A subscript adds a label to the brace, as in

$$\underbrace{a + a + \cdots + a}_{n}$$

which is typed as

```
\[
   \underbrace{a + a + \dots + a}_{n}
\]
```

The following example combines these two commands:

$$\underbrace{\overbrace{a + \cdots + a}^{(m-n)/2} + \underbrace{b + \cdots + b}_{n} + \overbrace{a + \cdots + a}^{(m-n)/2}}_{m}$$

This example is typed as

```
\[
   \underbrace{
      \overbrace{a + \dots + a}^{(m - n)/2}
      + \underbrace{b + \dots + b}_{n}
      + \overbrace{a + \dots + a}^{(m - n)/2}
    }_{m}
\]
```

3.9.2 Overlines and underlines:

The \overline and \underline commands draw lines above or below a formula. For example,

$$\overline{\overline{X} \cup \overline{\overline{X}}} = \overline{\overline{X}}$$

is typed as

```
\[
   \overline{ \overline{X} \cup \overline{\overline{X}} }
      = \overline{ \overline{X} }
\]
```

Similarly, you can place arrows above and below an expression:

$$\overleftarrow{a} \quad \overrightarrow{aa}$$

$$\overleftrightarrow{aaa} \quad \underleftarrow{aaaa} \quad \underrightarrow{aaaaa} \quad \underleftrightarrow{aaaaaa}$$

which is typed as

```
\begin{gather*}
   \overleftarrow{a}          \quad \overrightarrow{aa}\\
   \overleftrightarrow{aaa} \quad \underleftarrow{aaaa}\quad
   \underrightarrow{aaaaa}  \quad \underleftrightarrow{aaaaaa}
\end{gather*}
```

3.9.3 Stretchable arrow math symbols:

There are two stretchable arrow math symbols that extend to accommodate a formula above or below the arrows with the \xleftarrow and \xrightarrow commands. The formula on top is given as the argument (possibly empty) and the formula below is an optional argument.

$$A \xrightarrow{\text{1-1}} B \xleftarrow[\alpha\to\beta]{\text{onto}} C \xleftarrow[\gamma]{} D \xleftarrow{} E$$

is typed as

```
\[
    A \xrightarrow{\text{1-1}} B \xleftarrow[\alpha\to\beta]
       {\text{onto}} C \xleftarrow[\gamma]{} D \xleftarrow{} E
\]
```

3.10 Formula Gallery:

In this section present a collection of formulas—some simple, some complex—that illustrate the power of LATEX.

Some of these examples require the *amssymb* package, so it is a good idea to include the line

\usepackage{amssymb,latexsym}

following the *\documentclass* line of any article.

Formula 1: A set-valued function

$$x \mapsto \{c \in C \mid c \leq x\}$$

```
\[
   x \mapsto \{ c \in C \mid c \leq x \}
\]
```

Formula 2:

$$\left| \bigcup (I_j \mid j \in J) \right| < \mathfrak{m}$$

```
\[
   \left| \bigcup ( I_{j} \mid j \in J ) \right| < \mathfrak{m}
\]
```

Formula 3:

Note that we have to add spacing both before and after the text fragment in the following example. The argument of \text is typeset in text mode, so spaces are recognized.

$$A = \{x \in X \mid x \in X_i, \text{ for some } i \in I\}$$

```
\[
   A = \{ x \in X \mid x \in X_{i},
            \text{ for some $i \in I$} \}
\]
```

Formula 4:

Space to show logical structure

$$\langle a_1, a_2 \rangle \leq \langle a'_1, a'_2 \rangle \qquad \text{iff} \qquad a_1 < a'_1 \quad \text{or} \quad a_1 = a'_1 \text{ and } a_2 \leq a'_2$$

```
\[
   \langle a_{1}, a_{2} \rangle \leq
   \langle a'_{1}, a'_{2}\rangle \qquad \text{if{f}}
   \qquad a_{1} < a'_{1} \quad \text{or}
   \quad a_{1} = a'_{1} \text{ and } a_{2} \leq a'_{2}
\]
```

Note that in if{f} (in the argument of the first \text) the second f is enclosed in braces to avoid the use of the ligature—the merging of the two f's.

Formula 5:

Here are some examples of Greek letters

$$\Gamma_{u'} = \{\gamma \mid \gamma < 2\chi,\ B_\alpha \nsubseteq u',\ B_\gamma \subseteq u'\}$$

```
\[
   \Gamma_{u'} = \{\gamma \mid \gamma < 2\chi,\ B_{\alpha}
      \nsubseteq u', \ B_{\gamma} \subseteq u' \}
\]
```

Formula 6:

\mathbb allows you to use the blackboard bold math alphabet, which only provides capital letters

$$A = B^2 \times \mathbb{Z}$$

```
\[
   A = B^{2} \times \mathbb{Z}
\]
```

Formula 7:

\left[and \right] provide stretched delimiters

$$y^C \equiv z \vee \bigvee_{i \in C} \left[s_i^C \right] \pmod{\Phi}$$

```
\[
   y^C \equiv z \vee \bigvee_{ i \in C } \left[ s_{i}^{C}
     \right] \pmod{ \Phi }
\]
```

Formula 8:

A complicated congruence

$$y \vee \bigvee ([B_\gamma] \mid \gamma \in \Gamma) \equiv z \vee \bigvee ([B_\gamma] \mid \gamma \in \Gamma) \pmod{\Phi^x}$$

```
\[
   y \vee \bigvee ( [B_{\gamma}] \mid \gamma
    \in \Gamma ) \equiv z \vee \bigvee ( [B_{\gamma}]
    \mid \gamma \in \Gamma ) \pmod{ \Phi^{x} }
\]
```

Formula 9:

Use \nolimits to force the “limit” of the large operator to display as a subscript

$$f(\mathbf{x}) = \bigvee_{\mathfrak{m}} \left(\bigwedge_{\mathfrak{m}} (x_j \mid j \in I_i) \mid i < \aleph_\alpha \right)$$

```
\[
   f(\mathbf{x}) =
        \bigvee\nolimits_{\!\mathfrak{m}}
        \left(
       \bigwedge\nolimits_{\mathfrak{m}}
        ( x_{j} \mid j \in I_{i} )
        \mid i < \aleph_{\alpha}
        \right)
\]
```

Formula 10:

The \left. command gives a blank left delimiter, which is needed to balance the \right| command

$$\widehat{F}(x)\Big|_a^b = \widehat{F}(b) - \widehat{F}(a)$$

```
\[
   \left. \widehat{F}(x) \right|_{a}^{b}
    = \widehat{F}(b) - \widehat{F}(a)
\]
```

Unit-IV:

More math: Spacing of symbols building new symbols - math alphabets and symbols - vertical spacing - Tagging and grouping-Generalized Fractions - Boxed formulas.

4.0 More math:

Introduction:

In the previous chapter, we discussed the basic building blocks of a formula and how to put them together to form more complex formulas. This chapter starts out by going one step lower, to the characters that make up a formula. We discuss math symbols and math alphabets.

LATEX was designed for typesetting math, so it is not surprising that it contains a very large number of math symbols.

4.1 Spacing of symbols building new symbols:

LATEX provides a large variety of math symbols: Greek characters (α), binary operations ($\circ$), binary relations ($\leq$), negated binary relations ($\not\leq$), arrows ($\nearrow$), delimiters ({), and so on.

Consider the formula

$$A = \{x \in X \mid x\beta \geq xy > (x+1)^2 - \alpha\}$$

which is typed as

```
\[
   A = \{ x \in X \mid x \beta \geq x y
     > (x + 1)^{2} - \alpha \}
\]
```

The spacing of the symbols in the formula varies. In $x\beta$, the two symbols are very close. In $x \in X$, there is some space around the $\in$, and in $x + 1$, there is somewhat less space around the +.

4.1.1 Classification:

LATEX classifies symbols into several categories or *types* and spaces them accordingly. In the formula

$$A = \{x \in X \mid x\beta \geq xy > (x+1)^2 - \alpha\}$$

we find

- Ordinary math symbols: A, x, X, β, and so on
- Binary relations: =, ∈, | , ≥, and >
- Binary operations: + and −
- Delimiters: {, }, (, and)

As a rule, we do not have to be concerned with whether or not a given symbol in a formula, say ×, is a binary operation. LATEX knows and spaces the typeset symbol correctly.

4.1.2 Three exceptions:

There are three symbols with more than one classification:

$$+ \quad - \quad |$$

+ or − could be either a binary operation, for instance, $a - b$, or a sign, for instance $-b$.

4.1.3 Spacing commands:

There are some situations where LATEX cannot typeset a formula properly and we have to add spacing commands. Luckily, LATEX provides a variety of spacing commands, listed in Table 4.1. The \neg commands remove space by "reversing the print head".

The \quad and \qquad commands are often used to adjust aligned formulas or to add space before text in a math formula. The size of \quad (= 1 em) and \qquad (= 2 em) depends on the current font.

The \, and \! commands are the most useful for fine tuning math formulas. The \mspace command and the math unit *mu* provides us with even finer control. 18 mu = 1 em. For example, \mspace{3mu} adds a space that is 1/6 em long.

Name	Width	Short	Long
1 mu (math unit)	ı		\mspace{1mu}
thinspace	ıı	\,	\thinspace
medspace	ıı	\:	\medspace
thickspace	ıı	\;	\thickspace
interword space	ıı	\␣	
1 em	ı—ı		\quad
2 em	ı——ı		\qquad
Negative space			
1 mu	ı		\mspace{-1mu}
thinspace	ıı	\!	\negthinspace
medspace	ıı		\negmedspace
thickspace	ıı		\negthickspace

Table 4.1: Math spacing commands.

4.1.4 Examples:

We present some examples of fine tuning.

Example 1: We type the formula $\int_0^{\pi} sin\ x\ dx = 2$ as

```
$\int_{0}^{\pi} \sin x \, dx = 2$
```

Example 2 $|-f(x)|$, typed as `$|-f(x)|$`, is spaced incorrectly. $-$ becomes a binary operation by the $+$ and $-$ rule. To get the correct spacing, as in $|-f(x)|$, type `$\left|-f(x)\right|$`. This form tells LaTeX that the first | is a left delimiter, by the | rule, and therefore $-$ is the unary minus sign, not the binary subtraction operation.

Example 3 In $\sqrt{5}$side, typed as

```
$\sqrt{5} \text{side}$
```

$\sqrt{5}$ is too close to side. So type it as

```
$\sqrt{5} \, \text{side}$
```

which typesets as $\sqrt{5}\,$side.

Example 4 In $\sin x / \log n$, the division symbol / is too far from $\log n$, so type

```
$\sin x / \! \log n$
```

which prints $\sin x/\log n$.

Example 5 In $f(1/\sqrt{n})$, typed as

```
$f(1 / \sqrt{n})$
```

the square root almost touches the closing parenthesis. To correct it, type

```
$f(1 / \sqrt{n}\,)$
```

which typesets as $f(1/\sqrt{n}\,)$.

4.1.5 The phantom command:

The command produces a space in a formula equivalent to the space that would be occupied by its typeset argument. This command is one of the most powerful tools available to us for fine tuning alignments. Here are two simple illustrations:

$$A = \begin{pmatrix} 1 & 3 & 1 \\ 2 & 1 & 1 \\ -2 & 2 & -1 \end{pmatrix}$$

typed as

```
\[
   A = \begin{pmatrix}
          \phantom{-}1 & \phantom{-}3 & \phantom{-}1\\
          \phantom{-}2 & \phantom{-}1 & \phantom{-}1\\
                    -2 & \phantom{-}2 &           -1\\
       \end{pmatrix}
\]
```

and

$$a + b + c + d \quad = 0,$$
$$c + d + e = 5.$$

typed as

```
\begin{align*}
     a + b + c & + d \phantom{ {}+e } = 0,\\
             c & + d + e              = 5.
   \end{align*}
```

4.2 Math alphabets and symbols:

The symbols in a formula can also be classified as *characters from math alphabets* and *math symbols.* In the formula

$$A = \{x \in X \mid x\beta \geq xy > (x+1)^2 - \alpha\}$$

the following characters come from math alphabets:

$$A \quad x \quad X \quad y \quad 1 \quad 2$$

whereas these characters are math symbols:

$$= \quad \{ \quad \in \quad | \quad \beta \quad \geq \quad > \quad (\quad + \quad) \quad - \quad \alpha \quad \}$$

4.2.1 Math alphabets:

The letters and digits typed in a math formula come from *math alphabets.* LATEX's default math alphabet—the one we get if you do not ask for something else—is Computer Modern math italic for *letters.* In the formula $x^2 \vee y_3 = \alpha$, the characters x and y come from this math alphabet. The default math alphabet for *digits* is Computer Modern roman and the digits 2 and 3 in this formula are typeset in Computer Modern roman.

LATEX has a number of commands to switch type style in math. The two most important commands select the bold and italic versions:

Command	Math alphabet	Produces
`\mathbf{a}`	math bold	**2 Greek gammas, γ and Γ**
`\mathit{a}`	math italic	*2 Greek gammas, γ and Γ*

These commands change the style of letters, numbers, and upper case Greek characters. But beware of the pitfalls. For instance, in *\mathit{left-side}* the hyphen typesets as a minus: $left - side$.

There are four more commands that switch math alphabets:

Command	Math Alphabet	Produces
`\mathsf{a}`	math sans serif	2 Greek gammas, γ and Γ
`\mathrm{a}`	math roman	2 Greek gammas, γ and Γ
`\mathtt{a}`	math typewriter	2 Greek gammas, γ and Γ
`\mathnormal{a}`	math italic	*2 Greek gammas, γ and Γ*

Math roman is used in formulas for operator names, such as sin in $sin\ x$, and for text. For operator names, we should use the *\DeclareMathOperator* command or the *-ed version, which sets the name of the operator in math roman, and also provides the proper spacing.

The *\mathnormal* command switches to the default math alphabet; it is seldom used in practice.

The Computer Modern fonts include a math bold italic alphabet, but LATEX does not provide a command to access it.

4.2.2 Math symbol alphabets:

We may have noticed that α was not classified as belonging to an alphabet in the example at the beginning of this section. Indeed, α is treated by LATEX as a math symbol rather than as a member of a math alphabet. We cannot italicize or slant it, nor is there a sans serif version. There is a bold version, but you must use the \boldsymbol command to produce it. For instance, $\boldsymbol{\alpha}_{\boldsymbol{\beta}}$, is typed as

$\boldsymbol{\alpha}_{\boldsymbol{\beta}}$

Note that $\boldsymbol{\beta}$ appears in a small size in $\boldsymbol{\alpha}_{\boldsymbol{\beta}}$.

Four “alphabets of symbols” are built into LATEX.

Greek The examples α, β, Γ are typed as

α, β, Γ

Calligraphic an uppercase-only alphabet invoked with the \mathcal command.

The examples $\mathcal{A}, \mathcal{C}, \mathcal{E}$ are typed as

$\mathcal{A}, \mathcal{C}, \mathcal{E}$

Euler Fraktur invoked by the *\mathfrak* command. The examples $\mathfrak{n}, \mathfrak{N}, \mathfrak{p}, \mathfrak{P}$ are typed as

$\mathfrak{n}, \mathfrak{N}, \mathfrak{p}, \mathfrak{P}$

Blackboard bold uppercase-only math alphabet, invoked with *\mathbb*. The examples $\mathbb{A}, \mathbb{B}, \mathbb{C}$ are typed as

$\mathbb{A}, \mathbb{B}, \mathbb{C}$

4.3 Vertical spacing:

As a rule, all horizontal and vertical spacing in a math formula is done by LATEX. There is seldom a need to adjust vertical spacing, but there are a few exceptions.

The formula $\sqrt{a} + \sqrt{b}$ does not look quite right, because the square roots are not uniform. We can correct this with *\mathstrut* commands, which inserts an invisible vertical space:

$\sqrt{\mathstrut a} + \sqrt{\mathstrut b}$

typesets as $\sqrt{a} + \sqrt{b}$.

Another way to handle this situation is with the *\vphantom* (vertical phantom) command, which measures the height of its argument and places a math strut of that height into the formula. So

$\sqrt{\vphantom{b} a} + \sqrt{b}$

also prints uniform square roots, $\sqrt{a} + \sqrt{b}$. The \vphantom method is more versatile than the previous one.

Here is a more complicated example from a recent research article:

$$\Theta_i = \bigcup \left(\Theta(\overline{a \wedge b}, \overline{a} \wedge \overline{b}) \mid a,\ b \in B_i\right) \vee \bigcup \left(\Theta(\overline{a \vee b}, \overline{a} \vee \overline{b}) \mid a,\ b \in B_i\right),$$

typed as

```
\[
   \Theta_i = \bigcup \big( \Theta (\overline{a \wedge b},
   \overline{\vphantom{b}a} \wedge \overline{b})
   \mid a,\ b \in B_i \big)
   \vee \bigcup \big( \Theta(\overline{a \vee b},
   \overline{\vphantom{b}a} \vee \overline{b} )
   \mid a,\ b \in B_i \big),
 \]
```

Another useful command for vertical spacing is the \smash command. It directs LATEX to pretend that its argument does not protrude above or below the line in which it is typeset.

For instance, the two lines of this admonition:

It is *very important* that you memorize the integral $\frac{1}{\int f(x)\,dx} = 2g(x) + C$, which will appear on the next test.

are too far apart because LaTeX had to make room for the fraction. However, in this instance, the extra vertical space is not necessary because the second line is very short. To correct this, place the formula in the argument of a \smash command:

```
It is \emph{very important} that you memorize the
integral $\smash{\frac{1}{\int f(x) \, dx}} = 2g(x) + C$,
which will appear on the next test.
```

LaTeX produces the following:

It is *very important* that you memorize the integral $\frac{1}{\int f(x)\,dx} = 2g(x) + C$, which will appear on the next test.

An optional argument to the *smash* command controls which part of the formula is ignored, t to smash the top and b to smash the bottom.

4.4 Tagging and grouping:

We can attach a name to an equation using the \tag command. In the equation or equation* environments,

\tag{*name* }

attaches the tag *name* to the equation—*name* is typeset as text. The tag replaces the number.

Recall that the numbering of an equation is *relative,* that is, the number assigned to an equation is relative to the placement of the equation with respect to other equations in the document. An equation tag, on the other hand, is *absolute*—the tag remains the same even if the equation is moved.

If there is a tag, the equation and the equation* environments are equivalent.

For example,

(Int) $$\int_{-\infty}^{\infty} e^{-x^2}\,dx = \sqrt{\pi}$$

may be typed as

```
\begin{equation*}
   \int_{-\infty}^{\infty} e^{-x^{2}} \, dx
   = \sqrt{\pi}\tag{Int}
\end{equation*}
```

or

```
\begin{equation}
   \int_{-\infty}^{\infty} e^{-x^{2}} \, dx
   = \sqrt{\pi}\tag{Int}
\end{equation}
```

or

```
\[
   \int_{-\infty}^{\infty} e^{-x^{2}} \, dx
   = \sqrt{\pi}\tag{Int}
\]
```

The \tag* command is the same as \tag except that it does not automatically enclose the tag in parentheses. To get

A–B $$\int_{-\infty}^{\infty} e^{-x^2}\,dx = \sqrt{\pi}$$

type

```
\begin{equation}
  \int_{-\infty}^{\infty} e^{-x^{2}} \, dx = \sqrt{\pi}
   \tag*{A--B}
\end{equation}
```

In contrast, *grouping* applies to a group of *adjacent* equations. Suppose the last equation was numbered (1) and the next group of equations is to be referred to as (2), with individual equations numbered as (2a), (2b), and so on. Enclosing these equations in a subequations environment accomplishes this goal. For instance,

(1a) $$A^{[2]} \diamond B^{[2]} \cong (A \diamond B)^{[2]}$$

and its variant

(1b) $$A^{\langle 2 \rangle} \diamond B^{\langle 2 \rangle} \equiv (A \diamond B)^{\langle 2 \rangle}$$

are typed as

```
\begin{subequations}\label{E:joint}
   \begin{equation}\label{E:original}
      A^{[2]} \diamond B^{[2]} \cong (A \diamond B)^{[2]}
   \end{equation}

   \begin{equation}\label{E:modified}
      A^{\langle 2 \rangle} \diamond B^{\langle 2\rangle}
      \equiv (A \diamond B)^{\langle 2\rangle}
   \end{equation}
\end{subequations}
```

Referring to these equations, we find that

- \eqref{E:joint} resolves to (1)
- \eqref{E:original} resolves to (1a)
- \eqref{E:modified} resolves to (1b)

Note that in this example, references to the second and third labels produce numbers, (1a) and (1b), that also appear in the typeset version. The group label, E:joint, references the entire group, but (1) does not appear in the typeset version unless referenced.

4.5 Generalized Fractions:

The generalized fraction command provides the facility to typeset many variants of fractions and binomials, such as

$$\frac{a+b}{c} \text{ and } \left]{a+b \atop c}\right[.$$

The syntax is

\genfrac{left-delim }{right-delim }{thickness }{mathstyle }
{numerator }{denominator }

where

- *left-delim* is the left delimiter for the formula (default: none)
- *right-delim* is the right delimiter for the formula (default: none)
- *thickness* is the thickness of the fraction line, in the form *x* pt (default: the normal weight, 0.4pt), for instance, 12pt for 12 point width
- *mathstyle* is one of
  – 0 for \displaystyle
  – 1 for \textstyle
  – 2 for \scriptstyle
  – 3 for \scriptscriptstyle
  – Default: Depends on the context. If the formula is being set in display style, then the default is 0, and so on
- *numerator* is the numerator
- *denominator* is the denominator

All arguments must be specified. The empty argument, {}, gives the default value.

Examples

1. `\frac{`*`numerator`*`}{`*`denominator`*`}`
   is the same as

   `\genfrac{}{}{}{}{`*`numerator`*`}{`*`denominator`*`}`

2. `\dfrac{`*`numerator`*`}{`*`denominator`*`}`
   is the same as

   `\genfrac{}{}{}{0}{`*`numerator`*`}{`*`denominator`*`}`

3. `\tfrac{`*`numerator`*`}{`*`denominator`*`}`
   is the same as

   `\genfrac{}{}{}{1}{`*`numerator`*`}{`*`denominator`*`}`

4. `\binom{`*`numerator`*`}{`*`denominator`*`}`
   is the same as

   `\genfrac{(}{)}{0pt}{}{`*`numerator`*`}{`*`denominator`*`}`

5. Here are some more examples:

$$\frac{a+b}{c} \quad \genfrac{}{}{1pt}{}{a+b}{c} \quad \genfrac{}{}{1.5pt}{}{a+b}{c} \quad \genfrac{}{}{2pt}{}{a+b}{c} \quad \genfrac{[}{]}{0pt}{}{a+b}{c} \quad \genfrac{]}{[}{0pt}{}{a+b}{c}$$

   typed as

```
\[
   \frac{a + b}{c} \quad
   \genfrac{}{}{1pt}{}{a + b}{c}   \quad
   \genfrac{}{}{1.5pt}{}{a + b}{c} \quad
   \genfrac{}{}{2pt}{}{a + b}{c}   \quad
   \genfrac{[}{]}{0pt}{}{a + b}{c} \quad
   \genfrac{]}{[}{0pt}{}{a + b}{c}
\]
```

$$\frac{a+b}{c} \quad \genfrac{}{}{0.4pt}{}{a+b}{c}$$

   typed as

```
\[
   \frac{a + b}{c} \quad
   \genfrac{}{}{0.4pt}{}{a + b}{c}   \quad
\]
```

4.6 Boxed formulas:

The \boxed command puts its argument in a box, as in

$$\boxed{\int_{-\infty}^{\infty} e^{-x^2}\, dx = \sqrt{\pi}}$$

typed as

```
\begin{equation}
   \boxed{ \int_{-\infty}^{\infty} e^{-x^{2}}\, dx
   = \sqrt{\pi} }
\end{equation}
```

The \boxed command can also be used in the argument of a \text command. Note that

```
\fbox{Hello world}
```

and

```
$\boxed{\text{Hello world}}$
```

produce the same $\boxed{\text{Hello world}}$.

Morten Høgholm's mathtools package contains many variants of boxes.

Unit-V:

Latex documents: The structure of a document - The preamble - Abstract - Sectioning - Cross referencing - Bibliographies.

5.0 Latex documents:

Introduction:

In this chapter, we take up the organization of shorter documents, longer documents and books are discussed.

If we are writing a *simple article,* start with a template, then we can safely ignore much of the material discussed in this chapter. In more complicated articles we may need the material discussed in this chapter.

5.1 The structure of a document:

The source file of a LATEX document is divided into two main parts: the preamble and the body (see Figure 5.1).

Preamble This is the portion of the source file before the
\begin{document}
command. It contains definitions and instructions that affect the entire document.

Body This is the content of the document environment. It contains all the material to be typeset.

These statements over simplify the situation somewhat. For instance, we can define a command in the preamble to typeset some text that will appear wherever the command is used in the body, but the text is actually typed in the preamble. Nevertheless, we hope the division between the preamble and the body is clear

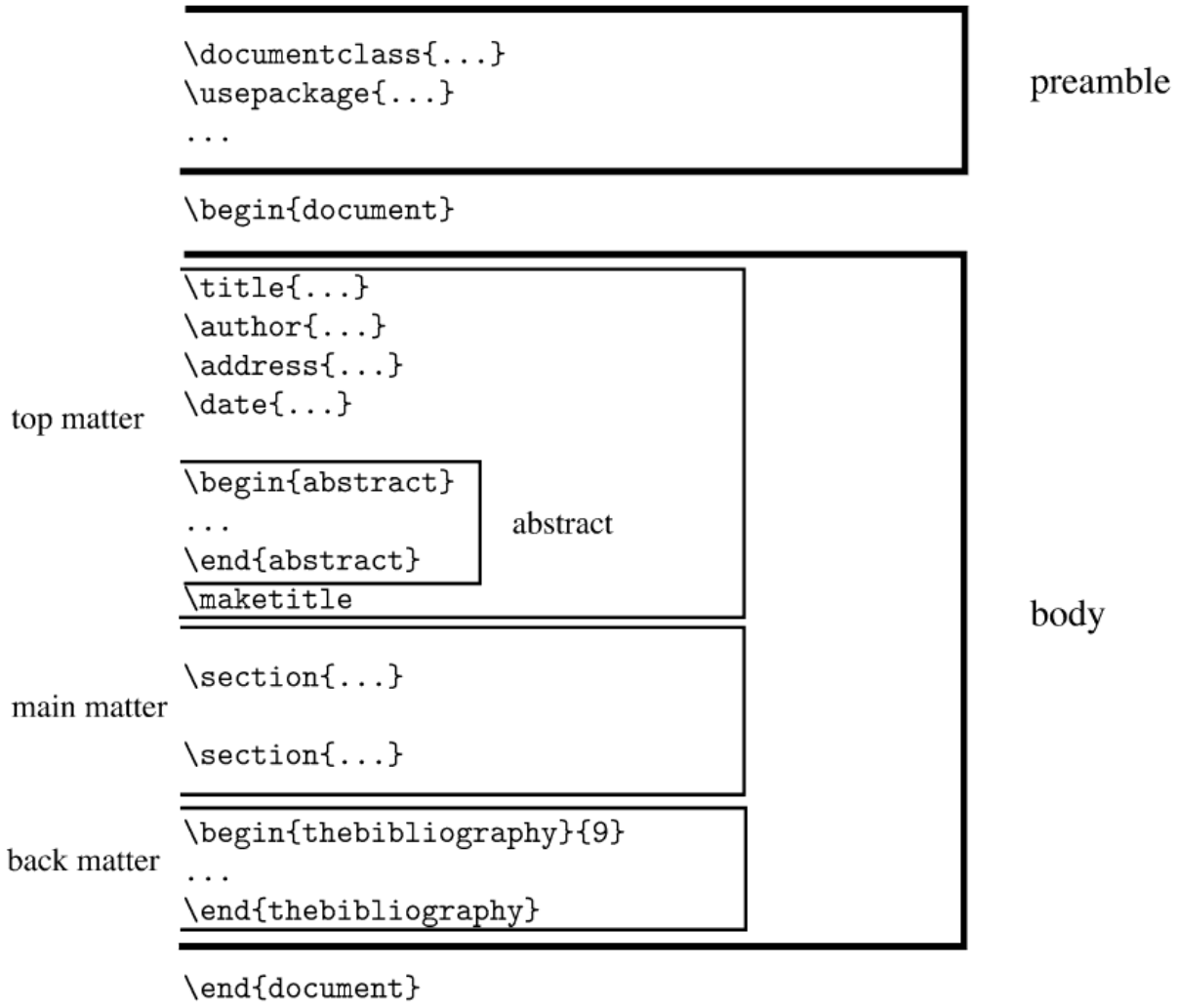

Figure 5.1: The structure of a LATEX document

The body is divided into three parts:

Top matter:

This is the first part of the body. It is concluded with the \maketitle command. Traditionally it included only the \title, the \author, and the \date commands. The top matter is derived from these commands and from it the title page of an article was designed. This evolved to include a lot more information about the author(s), for instance, their e-mail addresses, academic affiliations, home pages, and about the article, for instance, research support, subject classification. The typeset top matter now is split into several locations, the top and bottom of the first page and the bottom of the last page.

Main matter:

This is the main part of the document, including any appendices.

Back matter:

This is the material that is typeset at the end of the document. For a typical shorter document, the back matter is just the bibliography.

5.2 The preamble:

The preamble of a document contains the crucial \documentclass line, specifying the document class and the options that modify its behavior. For instance,

\documentclass[draft,reqno]{amsart}

loads the document class *amsart* with the draft option, which paints a slug in the margin indicating lines that are too wide, and the reqno option, which places the equation numbers on the right.

article is the most popular legacy document class. The command

\documentclass[titlepage,twoside]{article}

loads the document class article with the titlepage option, which creates a separate title page and places the abstract on a separate page, and the twoside option, which formats the typeset article for printing on both sides of the paper.

The \documentclass command is usually followed by the \usepackage commands, which load LATEX enhancements called *packages.* For instance,

\usepackage{latexsym}

loads a package that defines some additional LATEX symbol names, whereas loads the graphicx package with the demo option that inserts rectangles in place of the illustrations. Document class options are also passed on to the packages as possible options, so

\documentclass[demo]{amsart}

\usepackage{graphicx}

would also load the graphicx package with the demo option unless it is invoked with

\usepackage[final]{graphicx}

Any document class options that are not relevant for a package are ignored.

\usepackage commands can also be combined:

\usepackage{amssymb,latexsym}

is the same as

\usepackage{amssymb}

\usepackage{latexsym}

Document class files have a cls extension, whereas package files are designated by the sty extension. The document class amsart is defined in the amsart.cls file, the graphicx package is defined in the graphicx.sty file.

The preamble normally contains any custom commands and the proclamation definitions. Some commands can only be in the preamble. \DeclareMathOperator is such a command and so is \numberwithin. If we put such a command in the body, for example,

\DeclareMathOperator, you get a message:

! LaTeX Error: Can be used only in preamble.

l.103 \DeclareMathOperator

There is one command that may only be placed *before* the

\documentclass{...}

line:

\NeedsTeXFormat{LaTeX2e}[2005/12/01]

This command checks the version of LATEX being used to typeset the document and issues a warning if it is older than December 1, 2005 or whatever date you specified. Use this optional date argument if our document contains a feature that was introduced on or after the date specified or if an earlier version had a bug that would materially affect the typesetting of your document.

For instance, if we use the \textsubscript command, introduced in the December 1, 2005 release, then we may use the \NeedsTeXFormat line shown above. LATEX now hardly changes from year to year, so this command is rarely used except in document class files or package files.

5.3 Abstract:

Most standard document classes, except those for letters and books, make provision for an abstract, typed in an abstract environment.

The document class formats the heading as Abstract, or some variant, and, as a rule, typesets the text of the abstract in smaller type with wider margins.

The amsart document class requires that we place the abstract environment *before* the \maketitle command. If we forget to place it there, we get the warning

```
Class amsart Warning:
             Abstract should precede \maketitle in AMS
documentclasses; reported on input line 21.
```

and the abstract is typeset wherever the abstract environment happens to be placed.

In the article document class we place the abstract *after* the \maketitle command. If we place the abstract before the \maketitle command, the abstract is placed on page 1, and the article starts on page 2.

If the abstract and the "footnotes" from the top matter fill the first page, the second page has no running head. To fix this, follow the \maketitle command with the \clearpage command.

5.4 Sectioning:

The main matter of a typical shorter document is divided into *sections*.

Sections:

LATEX is instructed to start a section with the \section command, which takes the title of the section as its argument. This argument may also be used for the running head and it is also placed in the table of contents, which means that we need to protect fragile commands with the \protect command. LATEX automatically assigns a section number and typesets the section number followed by the section title.

Any \section command may be followed by a \label command, so that we can refer to the section number generated by LATEX, as in

\section{Introduction}\label{S:intro}

The command \ref{S:intro} refers to the number of the section and the command *\pageref{S:intro}* refers to the number of the typeset page where the section title appears.

We save a lot of work if in the source file you type in the cross-reference:

\section{Introduction}\label{S:intro}

%Section~\ref{S:intro}

Other sectioning commands:

A section may be subdivided into *subsections*, which may themselves be divided into *subsubsections*, *paragraphs*, and *subparagraphs.* Subsections are numbered within a section (in Section 1, they are numbered 1.1, 1.2, and so on). Here is the whole hierarchy:

```
\section
   \subsection
      \subsubsection
         \paragraph
            \subparagraph
```

It is important to understand that the five levels of sectioning are not just five different styles for typesetting section headers but they form a hierarchy. We should never have a subsection outside a section, a subsubsection outside a subsection, and so on. For instance, if the first sectioning command in our document is \subsection, the subsections are numbered 0.1, 0.2, ... Or if in the first section of your document the first sectioning command is \subsubsection, the subsubsections are numbered 1.0.1, 1.0.2, ... Both are clearly undesirable.

There are two additional sectioning commands provided by the report and by the book document classes *(book and amsbook): \chapter and \part.*

Any sectioning command may be followed by a \label command so that we can refer to the number (if any) generated by LATEX and the page on which it appears.

There is also the seldom used top level *\specialsection* command. Articles do not have parts and chapters, but sometimes a long article may require further division using the *\specialsection* command.

The form of sectioning commands:

All sectioning commands take one of the following three forms, illustrated below with the \section command:

Form 1:

The simplest form is

section{title }

where *title* is the section title, of course. We need to protect any fragile commands in *title* with the \protect command.

Form 2:

The sectioning command may have an optional argument

\section[*short_title*]{*title* }

The optional *short_title* argument is used in the running head. Protect any fragile commands in *short_title* with the \protect command.

Form 3:

Finally, we consider the *-ed version

\section*{*title* }

There are no section numbers printed and the *title* is not included in the running head. Remember that if we * a section, all subsections, and so on, must also be *-ed to avoid having strange section numbers.

Sectioning commands typeset:

Consider the following text:

```
\section{Introduction}\label{S:Intro}
We shall discuss the main contributors of this era.
\subsection{Birkhoff's contributions}\label{SS:contrib}
\subsubsection{The years 1935--1945}\label{SSS:1935}
Going to Oxford was a major step.
\paragraph{The first paper}
What should be the definition of a universal algebra?
\subparagraph{The idea}
One should read Whitehead very carefully.
```

This is how it looks typeset in the amsart document class:

1 Introduction

We shall discuss the main contributors of this era.

1.1 Birkhoff's contributions

1.1.1 The years 1935–1945

Going to Oxford was a major step.

The first paper What should be the definition of a universal algebra?

The idea One should read Whitehead very carefully.

Notice that paragraphs and subparagraphs are not displayed prominently by the AMS. By contrast, look at the same text typeset in the legacy article document class:

1. Introduction

We shall discuss the main contributors of this era.

1.1. Birkhoff's contributions.

1.1.1. *The years 1935–1945.* Going to Oxford was a major step.
The first paper. What should be the definition of a universal algebra?
The idea. One should read Whitehead very carefully.

This illustrates vividly one huge difference between the two document classes, the visual handling of sectioning.

5.5 Cross referencing:

There are three types of cross-referencing available in LATEX:

1. Symbolic referencing with \ref and \eqref for equations
2. Page referencing with \pageref
3. Bibliographic referencing with \cite

Symbolic referencing:

Wherever LATEX can automatically generate a number in your document, we can place a \label command

\label{*symbol* }

Then, at any place in our document, we can use the \ref command

\ref{*symbol* }

to place that number in the document. We call *symbol* the *label.* We can use labels for sectioning units, equations, figures, tables, items in an enumerated list environment, as well as for theorems and other proclamations.

If the equation labeled E:int is the fifth equation in an article, then LATEX stores the number 5 for the label E:int, so \ref{E:int} produces the number 5. If equations are numbered within sections, and an equation is the third equation in Section 2, then LATEX stores the number 2.3 for the label E:int, so the reference \ref{E:int} produces the number 2.3.

Example 1: The present section starts with the command

\section{Main matter}\label{S:MainMatter}

So \ref{S:MainMatter} produces the number 8.4 and we get the number of the typeset page where the section title appears with \pageref{S:MainMatter}, which is 234.

Example 2

```
\begin{equation}\label{E:int}
   \int_{0}^{\pi} \sin x \, dx = 2.
\end{equation}
```

In this case, \ref{E:int} produces the number of the equation, \eqref{E:int} produces the number of the equation in parentheses.

Example 3

```
\begin{theorem}\label{T:fund}
   Statement of theorem.
\end{theorem}
```

The reference \ref{T:fund} produces the number of the theorem.

Absolute referencing:

There are two forms of absolute referencing.

1. Equations can be tagged. The \tag{name } command attaches a name to the formula. The tag replaces the equation number.

2. Items in an itemize environment can be tagged with the \item[name] construct. The tag replaces the item number.

The first example is the equation

(Int) $$\int_0^{\pi} \sin x \, dx = 2$$

is typed as

```
\begin{equation}
  \int_{0}^{\pi} \sin x \, dx = 2 \tag{Int}
\end{equation}
```

The second example is the numbered list:

This space has the following properties:

(a) Grade 2 Cantor;
(b) Half-smooth Hausdorff;
(c) Metrizably smooth.

typed as

```
\noindent This space has the following properties:
\begin{enumerate}
   \item[(a)] Grade 2 Cantor\label{Cantor};
   \item[(b)] Half-smooth Hausdorff\label{Hausdorff};
   \item[(c)] Metrizably smooth\label{smooth}.
\end{enumerate}
```

Tags are *absolute*. This equation is *always* referred to as (Int). Equation numbers, on the other hand, are *relative*, they may change when the file is edited.

Page referencing:

The command

\pageref{symbol}

produces the number of the typeset page corresponding to the location of the command \label{symbol }. For example, if the following text is typeset on page 5,

```
There may be three types of problems with the
construction of such lattices.\label{problem}
```

and you type

```
Because of the problems associated with
the construction (see page~\pageref{problem})
```

anywhere in the document, LATEX produces

Because of the problems associated with the construction (see page 5)

5.6 Bibliographies in articles:

The simplest way to typeset a bibliography is to type it directly into the article. For an example, see the bibliography in the *secondarticle.tex* article.

The following bibliography contains two examples, one short and one long, of each of the seven most frequently used kinds of items.

We type the text of a bibliography in a *thebibliography* environment, as shown in the following examples.

```
\begin{thebibliography}{99}
\bibitem{hA70}
   Henry~H. Albert,
   \emph{Free torsoids},
   Current trends in lattice theory.
   D.~Van Nostrand, 1970.
\bibitem{hA70a}
```

```
   Henry~H. Albert,
   \emph{Free torsoids},
   Current trends in lattice theory
   (G.\,H. Birnbaum, ed.).
   vol.~7, D.~Van Nostrand, Princeton, January, 1970,
   no translation available, pp.~173--215 (German).
\bibitem{sF90}
   Soo-Key Foo,
   \emph{Lattice Constructions},
   Ph.D. thesis, University of Winnebago, 1990.
\bibitem{sF90a}
   Soo-Key Foo,
   \emph{Lattice Constructions},
   Ph.D. thesis, University of Winnebago, Winnebago, MN,
   December 1990, final revision not yet available.
\bibitem{gF86}
   Grant~H. Foster,
   \emph{Computational complexity in lattice theory},
   tech. report, Carnegie Mellon University, 1986.
\bibitem{gF86a}
   Grant~H. Foster,
   \emph{Computational complexity in lattice theory},
   Research Note 128A, Carnegie Mellon University,
   Pittsburgh, PA, December, 1986,
   research article in preparation.
\bibitem{pK69}
   Peter Konig,
   \emph{Composition of functions}.
   Proceedings of the Conference on Universal Algebra
   (Kingston, 1969).
\bibitem{pK69a}
   Peter Konig,
   \emph{Composition of functions}.
   Proceedings of the Conference on Universal Algebra
   (G.~H. Birnbaum, ed.).
   vol.~7, Canadian Mathematical Society,
   Queen's Univ., Kingston, ON,
   available from the Montreal office,
   pp.~1--106 (English).
\bibitem{wL75}
   William~A. Landau,
   \emph{Representations of complete lattices},
```

```
   Abstract: Notices Amer. Math. Soc. \textbf{18}, 937.
\bibitem{wL75a}
   William~A. Landau,
   \emph{Representations of complete lattices},
   Abstract: Notices Amer. Math. Soc. \textbf{18}, 937,
   December, 1975.
\bibitem{gM68}
   George~A. Menuhin,
   \emph{Universal algebra}.
   D.~Van Nostrand, Princeton, 1968.
\bibitem{gM68a}
   George~A. Menuhin,
   \emph{Universal algebra}. 2nd ed.,
   University Series in Higher Mathematics, vol.~58,
   D.~Van Nostrand, Princeton,
   March, 1968 (English), no Russian translation.
\bibitem{eM57}
   Ernest~T. Moynahan,
   \emph{On a problem of M. Stone},
   Acta Math. Acad. Sci. Hungar.
   \textbf{8}~(1957), 455--460.
\bibitem{eM57a}
   Ernest~T. Moynahan,
   \emph{On a problem of M. Stone},
   Acta Math. Acad. Sci. Hungar.
   \textbf{8}~(1957), 455--460
   (English), Russian translation available.
\end{thebibliography}
```

Figure 5.2 shows a typeset version of this bibliography in the *amsart* document class. By contrast, look at the same bibliography typeset in the legacy article document class in Figure 5.3.

We can find these entries in the document *inbibl.tpl* in the samples folder. We use the convention that the label for a \bibitem consists of the initials of the author and the year of publication. The first cited publication by Andrew B. Reich in 1987 would have the label aR87 and the second, aR87a. Of course, we can use any label we choose, but such conventions make the items easier to reuse.

The thebibliography environment takes an argument—in the previous example, this argument is 99—telling LATEX that the widest reference number it must generate is two digits wide. For fewer than 10 items, use 9 and for 100 or more items, use 999.

If the argument of \begin{thebibliography} is missing, we get the message

! LaTeX Error: Something's wrong--perhaps
a missing \item.

Each bibliographic item is introduced with \bibitem, which is used the same as the \label command. In our text, use \cite, in a similar way to \eqref—it provides the number enclosed in brackets. So if the 13th bibliographic item is introduced with

\bibitem{eM57}

then

\cite{eM57}

refers to that item and typesets it as [13]. The bibliography of the article itself is automatically numbered by LATEX. It is up to the author to make sure that the listing of the bibliographic items is in the proper order.

References

[1] Henry H. Albert, *Free torsoids*, Current trends in lattice theory. D. Van Nostrand, 1970.

[2] Henry H. Albert, *Free torsoids*, Current trends in lattice theory (G. H. Birnbaum, ed.). vol. 7, D. Van Nostrand, Princeton, January, 1970, no translation available, pp. 173–215 (German).

[3] Soo-Key Foo, *Lattice Constructions*, Ph.D. thesis, University of Winnebago, 1990.

[4] Soo-Key Foo, *Lattice Constructions*, Ph.D. thesis, University of Winnebago, Winnebago, MN, December 1990, final revision not yet available.

[5] Grant H. Foster, *Computational complexity in lattice theory*, tech. report, Carnegie Mellon University, 1986.

[6] Grant H. Foster, *Computational complexity in lattice theory*, Research Note 128A, Carnegie Mellon University, Pittsburgh, PA, December, 1986, research article in preparation.

[7] Peter Konig, *Composition of functions*. Proceedings of the Conference on Universal Algebra (Kingston, 1969).

[8] Peter Konig, *Composition of functions*. Proceedings of the Conference on Universal Algebra (G. H. Birnbaum, ed.). vol. 7, Canadian Mathematical Society, Queen's Univ., Kingston, ON, available from the Montreal office, pp. 1–106 (English).

[9] William A. Landau, *Representations of complete lattices*, Abstract: Notices Amer. Math. Soc. **18**, 937.

[10] William A. Landau, *Representations of complete lattices*, Abstract: Notices Amer. Math. Soc. **18**, 937, December, 1975.

[11] George A. Menuhin, *Universal algebra*. D. Van Nostrand, Princeton, 1968.

[12] George A. Menuhin, *Universal algebra*. 2nd ed., University Series in Higher Mathematics, vol. 58, D. Van Nostrand, Princeton, March, 1968 (English), no Russian translation.

[13] Ernest T. Moynahan, *On a problem of M. Stone*, Acta Math. Acad. Sci. Hungar. **8** (1957), 455–460.

[14] Ernest T. Moynahan, *On a problem of M. Stone*, Acta Math. Acad. Sci. Hungar. **8** (1957), 455–460 (English), Russian translation available.

Figure 5.2: The most important bibliographic entry types.

References

[1] Henry H. Albert, *Free torsoids*, Current trends in lattice theory, D. Van Nostrand, 1970.

[2] Henry H. Albert, *Free torsoids*, Current trends in lattice theory (G. H. Birnbaum, ed.), vol. 7, D. Van Nostrand, Princeton, January, 1970, no translation available, pp. 173–215 (German).

[3] Soo-Key Foo, *Lattice Constructions*, Ph.D. thesis, University of Winnebago, 1990.

[4] Soo-Key Foo, *Lattice Constructions*, Ph.D. thesis, University of Winnebago, Winnebago, MN, December 1990, final revision not yet available.

[5] Grant H. Foster, *Computational complexity in lattice theory*, tech. report, Carnegie Mellon University, 1986.

[6] Grant H. Foster, *Computational complexity in lattice theory*, Research Note 128A, Carnegie Mellon University, Pittsburgh, PA, December, 1986, research article in preparation.

[7] Peter Konig, *Composition of functions*. Proceedings of the Conference on Universal Algebra (Kingston, 1969).

[8] Peter Konig, *Composition of functions*. Proceedings of the Conference on Universal Algebra (G. H. Birnbaum, ed.), vol. 7, Canadian Mathematical Society, Queen's Univ., Kingston, ON, available from the Montreal office, pp. 1–106 (English).

[9] William A. Landau, *Representations of complete lattices*, Abstract: Notices Amer. Math. Soc., **18**, 937.

[10] William A. Landau, *Representations of complete lattices*, Abstract: Notices Amer. Math. Soc. **18**, 937, December, 1975.

[11] George A. Menuhin, *Universal algebra*. D. van Nostrand, Princeton, 1968.

[12] George A. Menuhin, *Universal algebra*. Second ed., University Series in Higher Mathematics, vol. 58, D. van Nostrand, Princeton, March, 1968 (English), no Russian translation.

[13] Ernest T. Moynahan, *On a problem of M. Stone*, Acta Math. Acad. Sci. Hungar. **8** (1957), 455–460.

[14] Ernest T. Moynahan, *On a problem of M. Stone*, Acta Math. Acad. Sci. Hungar. **8** (1957), 455–460 (English), Russian translation available.

Figure 5.3: Bibliography in the *article* document class.

- We can use \cite to cite two or more items in the form

\cite{hA70,eM57}

which typesets as [1, 13]. There is also an optional argument for \cite to specify additional information. For example,

\cite[pages~2--15]{eM57}

typesets as [13, pages 2–15].

- A label cannot contain a comma or a space.

Study Learning Material Prepared by

Dr. I. VALLIAMMAL, M.Sc., M.Phil., Ph.D.,

Assistant Professor, Department of Mathematics,

Manonmaniam Sundaranar University,Tirunelveli-12,

Tamil Nadu, India.