For us who wrote the following pages, \TeX{} is something beautiful and also useful. We enjoy \TeX{}, sharing the delights of newly discovered secrets amongst ourselves and wondering ever a new at the infinite variety of the program and the ingenuity of its creator. We also lend a helping hand to the new initiates to this art. Then we thought of extending this help to a wider group and The Net being the new medium, we started an online tutorial. This was well received and now the Free Software Foundation has decided to publish these lessons as a book. It is a fitting gesture that the organization which upholds the rights of the user to study and modify a software publish a book on one of the earliest programs which allows this right.

Dear reader, read the book, enjoy it and if possible, try to add to it.

The TUG\textit{India} Tutorial Team
# Contents

I. The Basics .......................................................... 7
   1.1 What is \LaTeX{}? – 7  1.2 Simple typesetting – 8  1.3 Fonts – 13  1.4 Type size – 15

II. The Document ..................................................... 17

III. Bibliography ..................................................... 27
   111.1 Introduction – 27  111.2 natbib – 28

IV. Bibliographic Databases .......................................... 33
   1111.1 The \BibTeX{} program – 33  1111.2 \BibTeX{} style files – 33  1111.3 Creating a bibliographic database – 34

V. Table of contents, Index and Glossary ........................... 39
   111.1 Table of contents – 39  111.2 Index – 41  111.3 Glossary – 44

VI. Displayed Text .................................................. 47
   1111.1 Borrowed words – 47  1111.2 Poetry in typesetting – 48  1111.3 Making lists – 48  1111.4 When order matters – 51  1111.5 Descriptions and definitions – 54

VII. Rows and Columns .............................................. 57
   1111.1 Keeping tabs – 57  1111.2 Tables – 62

VIII. Typesetting Mathematics ....................................... 77
   1111.1 The basics – 77  1111.2 Custom commands – 81  1111.3 More on mathematics – 82  1111.4 Mathematics miscellany – 89  1111.5 New operators – 101  1111.6 The many faces of mathematics – 102  1111.7 And that is not all! – 103  1111.8 Symbols – 103

IX. Typesetting Theorems ........................................... 109
   1111.1 Theorems in \LaTeX{} – 109  1111.2 Designer theorems—The \amsthm{} package – 111  1111.3 Housekeeping – 118

X. Several Kinds of Boxes .......................................... 119
   1111.1 LR boxes – 119  1111.2 Paragraph boxes – 121  1111.3 Paragraph boxes with specific height – 122  1111.4 Nested boxes – 125  1111.5 Rule boxes – 123

XI. Floats .................................................................. 125
   1111.1 The \figure{} environment – 125  1111.2 The \table{} environment – 130

5
XII. Cross References in \LaTeX{} .................................................. 135
  XII.1 Why cross references? – 135 • XII.2 Let \LaTeX{} do it – 135 •
  XII.3 Pointing to a page—the package \texttt{varioref} – 138 •
  XII.4 Pointing outside—the package \texttt{xt} – 140 •
  XII.5 Lost the keys? Use \texttt{lablst.tex} – 140

XIII. Footnotes, Marginpars, and Endnotes .......................... 143
  XIII.1 Footnotes – 143 • XIII.2 Marginal notes – 147 •
  XIII.3 Endnotes – 148
THE BASICS

1.1. What is \LaTeX?

The short and simple answer is that \LaTeX is a typesetting program and is an extension of the original program \TeX written by Donald Knuth. But then what is a *typesetting program?*

To answer this, let us look at the various stages in the preparation of a document using computers.

1. The text is *entered* into the computer.
2. The input text is *formatted* into lines, paragraphs and pages.
3. The output text is *displayed* on the computer screen.
4. The final output is *printed*.

In most *word processors* all these operations are integrated into a single application package. But a typesetting program like \TeX is concerned only with the second stage above. So to typeset a document using \TeX, we type the text of the document and the necessary formatting commands in a *text editor* (such as Emacs in GNU/Linux) and then compile it. After that the document can be viewed using a *previewer* or printed using a *printer driver*.

\TeX is also a *programming language*, so that by learning this language, people can write code for additional features. In fact \LaTeX itself is such a (large) collection of extra features. And the collective effort is continuing, with more and more people writing extra *packages*.

1.1.1. A small example

Let us see \LaTeX in action by typesetting a short (really short) document. Start your favorite text editor and type in the lines below *exactly* as shown

\begin{verbatim}
\documentclass{article}
\begin{document}
This is my \textit{first} document prepared in \LaTeX.
\end{document}
\end{verbatim}

Be especially careful with the \ character (called the *backslash*) and note that this is different from the more familiar / (the *slash*) in *and/or* and save the file onto the hard disk as myfile.tex. (Instead of myfile you can use any name you wish, but be sure to have .tex at the end as the *extension.*) The process of compiling this and viewing the output depends on your operating system. We describe below the process of doing this in GNU/Linux.
At the shell prompt type
\latex\ myfile

You will see a number of lines of text scroll by in the screen and then you get the prompt back. To view the output in screen, you must have the X Window running. So, start X if you have not done so, and in a terminal window, type
\xdvi\ myfile

A window comes up showing the output below

This is my first document prepared in \LaTeX.

Now let us take a closer look at the source file (that is, the file you have typed). The first line \documentclass{article} tells \LaTeX\ that what we want to produce is an article. If you want to write a book, this must be changed to \documentclass{book}.

The whole document we want to typeset should be included between \begin{document} and \end{document}. In our example, this is just one line. Now compare this line in the source and the output. The first three words are produced as typed. Then \emph{first}, becomes first in the output (as you have probably noticed, it is a common practice to \textit{emphasize} words in print using italic letters). Thus \emph is a command to \LaTeX\ to typeset the text within the braces in \textit{italic}'. Again, the next three words come out without any change in the output. Finally, the input \LaTeX\ comes out in the output as \LaTeX.

Thus our source is a mixture of text to be typeset and a couple of \LaTeX\ commands \emph and \LaTeX. The first command changes the input text in a certain way and the second one generates new text. Now call up the file again and add one more sentence given below.

This is my \emph{first} document prepared in \LaTeX. I typed it on \today.

What do you get in the output? What new text does the command \today generate?

1.1.2. Why \LaTeX?  

So, why all this trouble? Why not simply use a word processor? The answer lies in the motivation behind \TeX. Donald Knuth says that his aim in creating \TeX is to \textit{beautifully} typeset \textit{technical documents} especially those containing \textit{a lot of Mathematics}. It is very difficult (sometimes even impossible) to produce complex mathematical formulas using a word processor. Again, even for ordinary text, if you want your document to look \textit{really beautiful} then \LaTeX is the natural choice.

1.2. SIMPLE TYPESETTING

We have seen that to typeset something in \LaTeX, we type in the text to be typeset together with some \LaTeX commands. Words must be separated by spaces (does not matter how many) and lines maybe broken arbitrarily.

The end of a paragraph is specified by a \textit{blank line} in the input. In other words, whenever you want to start a new paragraph, just leave a blank line and proceed. For example, the first two paragraphs above were produced by the input

\footnote{This is not really true. For the real story of the \texttt{command}, see the section on fonts.}
We have seen that to typeset something in \LaTeX, we type in the text to be typeset together with some \LaTeX\ commands. Words must be separated by spaces (does not matter how many) and lines maybe broken arbitrarily.

The end of a paragraph is specified by a blank line in the input. In other words, whenever you want to start a new paragraph, just leave a blank line and proceed.

Note that the first line of each paragraph starts with an indentation from the left margin of the text. If you do not want this indentation, just type \texttt{\noindent} at the start of each paragraph for example, in the above input, \texttt{\noindent We have seen ...} and \texttt{\noindent The end of ...} (come on, try it!) There is an easier way to suppress paragraph indentation for all paragraphs of the document in one go, but such tricks can wait.

### 1.2.1. Spaces

You might have noticed that even though the length of the lines of text we type in a paragraph are different, in the output, all lines are of equal length, aligned perfectly on the right and left. \TeX\ does this by adjusting the space between the words.

In traditional typesetting, a little extra space is added to periods which end sentences and \TeX\ also follows this custom. But how does \TeX\ know whether a period ends a sentence or not? It assumes that every period not following an upper case letter ends a sentence. But this does not always work, for there are instances where a sentence does end in an upper case letter. For example, consider the following:

```
Carrots are good for your eyes, since they contain Vitamin A. Have you ever seen a rabbit wearing glasses?
```

The right input to produce this is

```
Carrots are good for your eyes, since they contain Vitamin A@. Have you ever seen a rabbit wearing glasses?
```

Note the use of the command \texttt{@} before the period to produce the extra space after the period. (Remove this from the input and see the difference in the output.)

On the other hand, there are instances where a period following a lowercase letter does not end a sentence. For example

```
The numbers 1, 2, 3, etc. are called natural numbers. According to Kronecker, they were made by God; all else being the work of Man.
```

Here, we use the command \texttt{\space} (that is, a backslash and a space—here and elsewhere, we sometimes use \texttt{\space} to denote a space in the input, especially when we draw attention to the space).

There are other situations where the command \texttt{\space} (which always produce a space in the output) is useful. For example, type the following line and compile it.

```
I think \LaTeX\ is fun.
```
You get

```
I think \LaTeX is fun.
```

What happened to the space you typed between \LaTeX and is? You see, \TeX gobbles up all spaces after a command. To get the required sequence in the output, change the input as

```
I think \LaTeX is fun.
```

Again, the command \ comes to the rescue.

### 1.2.2. Quotes

Have you noticed that in typesetting, opening quotes are different from closing quotes? Look at the \TeX output below

```
Note the difference in right and left quotes in 'single quotes' and “double quotes”.
```

This is produced by the input

```
Note the difference in right and left quotes in 'single quotes' and ‘double quotes’.
```

Modern computer keyboards have a key to type the symbol \` which produces a left quote in \TeX. (In our simulated inputs, we show this symbol as ‘.) Also, the key ’ (the usual ‘typewriter’ quote key, which also doubles as the apostrophe key) produces a left quote in \TeX. Double quotes are produced by typing the corresponding single quote twice. The ‘usual’ double quote key “ can also be used to produce a closing double quote in \TeX.

If your keyboard does not have a left quote key, you can use \lq command to produce it. The corresponding command \rq produces a right quote. Thus the output above can also be produced by

```
Note the difference in right and left quotes in \lq single quotes\rq and \lq\lq double quotes\rq\rq.
```

(Why the command \ after the first \rq?)

### 1.2.3. Dashes

In text, dashes are used for various purposes and they are distinguished in typesetting by their lengths; thus short dashes are used for hyphens, slightly longer dashes are used to indicate number ranges and still longer dashes used for parenthetical comments. Look at the following \TeX output

```
X-rays are discussed in pages 221–225 of Volume 3—the volume on electromagnetic waves.
```

This is produced from the input

```
X-rays are discussed in pages 221--225 of Volume 3---the volume on electromagnetic waves.
```

Note that a single dash character in the input -- produces a hyphen in the output, two dashes -- produces a longer dash (–) in the output and three dashes --- produce the longest dash (—) in the output.
1.2.4. Accents

Sometimes, especially when typing foreign words in English, we need to put different types of accents over the letters. The table below shows the accents available in \LaTeX. Each column shows some of the accents and the inputs to generate them.

<table>
<thead>
<tr>
<th>ò</th>
<th>ø</th>
<th>ò</th>
<th>ö</th>
<th>ò</th>
<th>¨ó</th>
</tr>
</thead>
<tbody>
<tr>
<td>ò</td>
<td>ò</td>
<td>ò</td>
<td>ò</td>
<td>ò</td>
<td>ç</td>
</tr>
</tbody>
</table>

The letters i and j need special treatment with regard to accents, since they should not have their customary dots when accented. The commands \i and \j produce dot-less i and j as í and j. Thus to get

\texttt{Él està aquí}

you must type

\texttt{\{É\}l est\{a\} aqu\{i\}}

Some symbols from non-English languages are also available in \LaTeX, as shown in the table below:

<table>
<thead>
<tr>
<th>œ</th>
<th>œ</th>
<th>Œ</th>
<th>œ</th>
</tr>
</thead>
<tbody>
<tr>
<td>ø</td>
<td>Ø</td>
<td>ò</td>
<td>ò</td>
</tr>
<tr>
<td>œ</td>
<td>œ</td>
<td>œ</td>
<td>œ</td>
</tr>
<tr>
<td>œ</td>
<td>œ</td>
<td>œ</td>
<td>œ</td>
</tr>
<tr>
<td>œ</td>
<td>œ</td>
<td>œ</td>
<td>œ</td>
</tr>
</tbody>
</table>

1.2.5. Special symbols

We have seen that the input \LaTeX produces \LaTeX in the output and \ produces a space. Thus \TeX uses the symbol \ for a special purpose—to indicate the program that what follows is not text to be typeset but an instruction to be carried out. So what if you want to get \ in your output (improbable as it may be)? The command \textbackslash produces \ in the output.

Thus \ is a symbol which has a special meaning for \TeX and cannot be produced by direct input. As another example of such a special symbol, see what is obtained from the input below

\texttt{Maybe I have now learnt about 1\% of \LaTeX.}

You only get

\texttt{Maybe I have now learnt about 1}

What happened to the rest of the line? You see, \TeX uses the per cent symbol % as the comment character; that is a symbol which tells \TeX to consider the text following as ‘comments’ and not as text to be typeset. This is especially useful for a \TeX programmer to explain a particularly sticky bit of code to others (and perhaps to himself). Even for ordinary users, this comes in handy, to keep a ‘to do’ list within the document itself for example.

But then, how do you get a percent sign in the output? Just type \% as in
Maybe I have now learnt about 1% of LaTeX.

The symbols \ and % are just two of the ten characters \TeX{} reserves for its internal use. The complete list is

\[ \tilde{\textasciitilde} \# \$ \% \& _ { \{ \} } \]

We have seen how \TeX{} uses two of these symbols (or is it four? Did not we use { } in one of our examples?) The use of others we will see as we proceed.

Also, we have noted that \ is produced in the output by the command \textbackslash{} and % is produced by \%. What about the other symbols? The table below gives the inputs to produce these symbols.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>\tilde{\textasciitilde}</td>
<td>\textasciitilde</td>
</tr>
<tr>
<td>#</td>
<td>#</td>
</tr>
<tr>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td>%</td>
<td>%</td>
</tr>
<tr>
<td>^</td>
<td>\textasciicircum</td>
</tr>
</tbody>
</table>

You can see that except for three, all special symbols are produced by preceding them with a \. Of the exceptional three, we have seen that \ and \ are used for producing accents. So what does \ do? It is used to break lines. For example,

This is the first line.\ This is the second line

produces

This is the first line.
This is the second line

We can also give an optional argument to \ to increase the vertical distance between the lines. For example,

This is the first line.\[10pt]\ This is the second line

This is the first line.
This is the second line

Now there is an extra 10 points of space between the lines (1 point is about \(\frac{1}{72}\)th of an inch).

1.2.6. Text positioning

We have seen that \TeX{} aligns text in its own way, regardless of the way text is formatted in the input file. Now suppose you want to typeset something like this

The \TeX{}nical Institute

Certificate

This is to certify that Mr. N. O. Vice has undergone a course at this institute and is qualified to be a \TeX{}nician.

The Director
The \TeX{}nical Institute

This is produced by
I.3. Fonts

The \TeX\, nical Institute

Certificate

This is to certify that Mr. N. O. Vice has undergone a course at this institute and is qualified to be a \TeX\, nician.

The Director

The \TeX\, nical Institute

Here, the commands
\begin{center} ... \end{center}

typesets the text between them exactly at the center of the page and the commands
\begin{flushright} ... \end{flushright}

typesets text flush with the right margin. The corresponding commands
\begin{flushleft} ... \end{flushleft}
places the enclosed text flush with the left margin. (Change the flushright to flushleft and see what happens to the output.)

These examples are an illustration of a \LaTeX\ construct called an environment, which is of the form
\begin{name} ... \end{name}

where name is the name of the environment. We have seen an example of an environment at the very beginning of this chapter (though not identified as such), namely the document environment.

I.3. Fonts

The actual letters and symbols (collectively called type) that \LaTeX\ (or any other typesetting system) produces are characterized by their style and size. For example, in this book emphasized text is given in italic style and the example inputs are given in typewriter style. We can also produce smaller and bigger type. A set of types of a particular style and size is called a font.

1.3.1. Type style

In \LaTeX, a type style is specified by family, series and shape. They are shown in the table 1.1.

Any type style in the output is a combination of these three characteristics. For example, by default we get roman family, medium series, upright shape type style in a \LaTeX\ output. The \textit\ command produces roman family, medium series, italic shape type. Again, the command \textbf\ produces roman family, boldface series, upright shape type.

We can combine these commands to produce a wide variety of type styles. For example, the input
\textsf{\textbf{sans serif family, boldface series, upright shape}}
\texttt{\textsl{roman family, medium series, slanted shape}}
produces the output shown below:

<table>
<thead>
<tr>
<th>sans serif family, boldface series, upright shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>roman family, medium series, slanted shape</td>
</tr>
</tbody>
</table>

Some of these type styles may not be available in your computer. In that case, \LaTeX\ gives a warning message on compilation and substitutes another available type style which it thinks is a close approximation to what you had requested.

We can now tell the whole story of the \textit{} command. We have seen that it usually, that is when we are in the middle of normal (upright) text, it produces italic shape. But if the current type shape is slanted or italic, then it switches to upright shape. Also, it uses the family and series of the current font. Thus
\begin{verbatim}
\textit{A polygon of three sides is called a \textbf{triangle} and a polygon of four sides is called a \textit{quadrilateral}}
\end{verbatim}
gives

\begin{center}
A polygon of three sides is called a triangle and a polygon of four sides is called a quadrilateral
\end{center}

while the input
\begin{verbatim}
\textbf{A polygon of three sides is called a \textit{triangle} and a polygon of four sides is called a \textbf{quadrilateral}}
\end{verbatim}
produces

\begin{center}
A polygon of three sides is called a \textbf{triangle} and a polygon of four sides is called a \textit{quadrilateral}
\end{center}

Each of these type style changing commands has an alternate form as a declaration. For example, instead of \textbf{boldface} you can also type \textbfseries boldface to get boldface. Note that that not only the name of the command, but its usage also is different. For example, to typeset
By a triangle, we mean a polygon of three sides.

if you type

By a \textbf{triangle}, we mean a polygon of three sides.

you will end up with

By a \textbf{triangle}, we mean a polygon of three sides.

Thus to make the declaration act upon a specific piece of text (and no more), the declaration and the text should be enclosed in braces.

The table below completes the one given earlier, by giving also the declarations to produce type style changes.

<table>
<thead>
<tr>
<th>Style</th>
<th>Command</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>upright</td>
<td>\textup{}</td>
<td>\textupshape upright</td>
</tr>
<tr>
<td>italic</td>
<td>\textit{}</td>
<td>\itshape italic</td>
</tr>
<tr>
<td>slanted</td>
<td>\textsl{}</td>
<td>\slshape slanted</td>
</tr>
<tr>
<td>small cap</td>
<td>\textsc{}</td>
<td>\scshape small cap</td>
</tr>
<tr>
<td>medium</td>
<td>\textmd{}</td>
<td>\mdseries medium</td>
</tr>
<tr>
<td>boldface</td>
<td>\textbf{}</td>
<td>\bfseries boldface</td>
</tr>
<tr>
<td>roman</td>
<td>\textsf{}</td>
<td>\sfseries sans serif</td>
</tr>
<tr>
<td>sans serif</td>
<td>\texttt{}</td>
<td>\ttfamily typewriter</td>
</tr>
</tbody>
</table>

These declaration names can also be used as environment names. Thus to typeset a long passage in, say, sans serif, just enclose the passage within the commands \begin{sffamily} ... \end{sffamily}.

### 1.4. Type size

Traditionally, type size is measured in (printer) points. The default type that \TeX produces is of 10 pt size. There are some \textit{declarations} (ten, to be precise) provided in \LaTeX for changing the type size. They are given in the following table:

<table>
<thead>
<tr>
<th>size</th>
<th>\tiny size</th>
<th>size</th>
<th>\large size</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>\scriptsize size</td>
<td>size</td>
<td>\Large size</td>
</tr>
<tr>
<td>size</td>
<td>\footnotesize size</td>
<td>size</td>
<td>\LARGE size</td>
</tr>
<tr>
<td>size</td>
<td>\small size</td>
<td>size</td>
<td>\huge size</td>
</tr>
<tr>
<td>size</td>
<td>\normalsize size</td>
<td>size</td>
<td>\Huge size</td>
</tr>
</tbody>
</table>

Note that the \texttt{\normalsize} corresponds to the size we get by default and the sizes form an ordered sequence with \tiny producing the smallest and \Huge producing the largest. Unlike the style changing commands, there are no \texttt{command-with-one-argument} forms for these declarations.

We can combine style changes with size changes. For example, the “certificate” we typed earlier can now be ‘improved’ as follows

\begin{center}
\textbf{Huge The \TeX nical Institute}\\[1cm]
\textsc{LARGE Certificate}
\end{center}
This is to certify that Mr. N. O. Vice has undergone a course at this institute and is qualified to be a \TeX nical Expert.

\begin{flushright}
\text{sffamily The Director
The \TeX nical Institute}
\end{flushright}

and this produces

\begin{center}
\noindent This is to certify that Mr. N. O. Vice has undergone a course at this institute and is qualified to be a \TeX nical Expert.
\end{center}

The \TeX nical Institute

CERTIFICATE

This is to certify that Mr. N. O. Vice has undergone a course at this institute and is qualified to be a \TeX nical Expert.

The Director
The \TeX nical Institute
II. DOCUMENT CLASS

We now describe how an entire document with chapters and sections and other embellishments can be produced with \LaTeX. We have seen that all \LaTeX files should begin by specifying the kind of document to be produced, using the command \texttt{\documentclass{...}}. We've also noted that for a short article (which can actually turn out to be quite long!) we write \texttt{\documentclass{article}} and for books, we write \texttt{\documentclass{book}}. There are other document classes available in \LaTeX such as \texttt{report} and \texttt{letter}. All of them share some common features and there are features specific to each.

In addition to specifying the type of document (which we \textit{must} do, since \LaTeX has no default document class), we \textit{can} also specify some options which modify the default format. Thus the actual syntax of the \texttt{\documentclass} command is

\begin{verbatim}
\documentclass[options]{class}
\end{verbatim}

Note that options are given in \textit{square brackets} and not braces. (This is often the case with \LaTeX commands—options are specified within square brackets, after which mandatory arguments are given within braces.)

II.1. Font size

We can select the size of the font for the normal text in the entire document with one of the options

\begin{itemize}
\item 10pt
\item 11pt
\item 12pt
\end{itemize}

Thus we can say

\begin{verbatim}
\documentclass[11pt]{article}
\end{verbatim}

to set the normal text in our document in 11 pt size. The default is 10pt and so this is the size we get, if we do not specify any font-size option.

II.1.2. Paper size

We know that \LaTeX has its own method of breaking lines to make paragraphs. It also has methods to make vertical breaks to produce different pages of output. For these breaks to work properly, it must know the width and height of the paper used. The various options for selecting the paper size are given below:

\begin{center}
\begin{tabular}{|l|c|c|}
\hline
\texttt{letterpaper} & 11\times8.5 in & \texttt{a4paper} & 20.7\times28 in \\
\texttt{legalpaper} & 14\times8.5 in & \texttt{a5paper} & 21\times14.8 in \\
\texttt{executivepaper} & 10.5\times7.25 in & \texttt{b5paper} & 25\times17.6 in \\
\hline
\end{tabular}
\end{center}

Normally, the longer dimension is the vertical one—that is, the height of the page. The default is \texttt{letterpaper}.
II.1.3. Page formats

There are options for setting the contents of each page in a single column (as is usual) or in two columns (as in most dictionaries). This is set by the options

onecolumn  twocolumn

and the default is onecolumn.

There is also an option to specify whether the document will be finally printed on just one side of each paper or on both sides. The names of the options are

oneside  twoside

One of the differences is that with the twoside option, page numbers are printed on the right on odd-numbered pages and on the left on even numbered pages, so that when these printed back to back, the numbers are always on the outside, for better visibility. (Note that \LaTeX{} has no control over the actual printing. It only makes the formats for different types of printing.) The default is oneside for article, report and letter and twoside for book.

In the report and book class there is a provision to specify the different chapters (we will soon see how). Chapters always begin on a new page, leaving blank space in the previous page, if necessary. With the book class there is the additional restriction that chapters begin only on odd-numbered pages, leaving an entire page blank, if need be. Such behavior is controlled by the options,

openany  openright

The default is openany for reportclass (so that chapters begin on “any” new page) and openright for the book class (so that chapters begin only on new right, that is, odd numbered, page).

There is also a provision in \LaTeX{} for formatting the “title” (the name of the document, author(s) and so on) of a document with special typographic consideration. In the article class, this part of the document is printed along with the text following on the first page, while for report and book, a separate title page is printed. These are set by the options

notitlepage  titlepage

As noted above, the default is notitlepage for article and titlepage for report and book. As with the other options, the default behavior can be overruled by explicitly specifying an option with the documentclass command.

There are some other options to the documentclass which we will discuss in the relevant context.

II.2. Page style

Having decided on the overall appearance of the document through the \documentclass command with its various options, we next see how we can set the style for the individual pages. In \LaTeX{} parlance, each page has a “head” and “foot” usually containing such information as the current page number or the current chapter or section. Just what goes where is set by the command

\pagestyle{...}

where the mandatory argument can be any one of the following styles

plain  empty  headings  myheadings

The behavior pertaining to each of these is given below:
The page head is empty and the foot contains just the page number, centered with respect to the width of the text. This is the default for the article class if no \pagestyle is specified in the preamble.

Both the head and foot are empty. In particular, no page numbers are printed.

This is the default for the book class. The foot is empty and the head contains the page number and names of the chapter section or subsection, depending on the document class and its options as given below:

<table>
<thead>
<tr>
<th>CLASS</th>
<th>OPTION</th>
<th>LEFT PAGE</th>
<th>RIGHT PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>book, report</td>
<td>one-sided</td>
<td>—</td>
<td>chapter</td>
</tr>
<tr>
<td></td>
<td>two-sided</td>
<td>chapter</td>
<td>section</td>
</tr>
<tr>
<td>article</td>
<td>one-sided</td>
<td>—</td>
<td>section</td>
</tr>
<tr>
<td></td>
<td>two-sided</td>
<td>section</td>
<td>subsection</td>
</tr>
</tbody>
</table>

The same as headings, except that the ‘section’ information in the head are not predetermined, but to be given explicitly using the commands \markright or \markboth as described below.

Moreover, we can customize the style for the current page only using the command

\thispagestyle{style}

where style is the name of one of the styles above. For example, the page number may be suppressed for the current page alone by the command \thispagestyle{empty}. Note that only the printing of the page number is suppressed. The next page will be numbered with the next number and so on.

As we mentioned above, in the page style myheadings, we have to specify the text to appear on the head of every page. It is done with one of the commands

\markboth{left head}{right head}
\markright{right head}

where left head is the text to appear in the head on left-hand pages and right head is the text to appear on the right-hand pages.

The \markboth command is used with the twoside option with even numbered pages considered to be on the left and odd numbered pages on the right. With oneside option, all pages are considered to be right-handed and so in this case, the command \markright can be used. These commands can also be used to override the default head set by the headings style.

Note that these give only a limited control over the head and foot. since the general format, including the font used and the placement of the page number, is fixed by \LaTeX. Better customization of the head and foot are offered by the package \fancyhdr, which is included in most \LaTeX distributions.

The style of page numbers can be specified by the command

\pagenumbering{...}

The possible arguments to this command and the resulting style of the numbers are given below:
II. THE DOCUMENT

arabic  Indo-Arabic numerals
roman  lowercase Roman numerals
Roman  upper case Roman numerals
alph  lowercase English letters
Alph  uppercase English letters

The default value is arabic. This command resets the page counter. Thus for example, to number all the pages in the ‘Preface’ with lowercase Roman numerals and the rest of the document with Indo-Arabic numerals, declare \pagenumbering{roman} at the beginning of the Preface and issue the command \pagestyle{arabic} immediately after the first \chapter command. (The \chapter{...} command starts a new chapter. We will come to it soon.)

We can make the pages start with any number we want by the command
\setcounter{page}{number}
where number is the page number we wish the current page to have.

II.4. FORMATTING LENGTHS

Each page that \LaTeX produces consists not only of a head and foot as discussed above but also a body (surprise!) containing the actual text. In formatting a page, \LaTeX uses the width and heights of these parts of the page and various other lengths such as the left and right margins. The values of these lengths are set by the paper size options and the page format and style commands. For example, the page layout with values of these lengths for an odd page and even in this book are separately shown below.

These lengths can all be changed with the command \setlength. For example,
\setlength{\textwidth}{15cm}
makes the width of text 15 cm. The package geometry gives easier interfaces to customize page format.

II.5. PARTS OF A DOCUMENT

We now turn our attention to the contents of the document itself. Documents (especially longer ones) are divided into chapters, sections and so on. There may be a title part (sometimes even a separate title page) and an abstract. All these require special typographic considerations and \LaTeX has a number of features which automate this task.

II.5.1. Title

The “title” part of a document usually consists of the name of the document, the name of author(s) and sometimes a date. To produce a title, we make use of the commands
\title{document name}
\author{author names}
\date{date text}
\maketitle

Note that after specifying the arguments of \title, \author and \date, we must issue the command \maketitle for this part to be typeset.

By default, all entries produced by these commands are centered on the lines in which they appear. If a title text is too long to fit in one line, it will be broken automatically. However, we can choose the break points with the \ command.

If there are several authors and their names are separated by the \ and command, then the names appear side by side. Thus
produces

<table>
<thead>
<tr>
<th>Author 1</th>
<th>Author 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address line 11</td>
<td>Address line 21</td>
</tr>
<tr>
<td>Address line 12</td>
<td>Address line 22</td>
</tr>
<tr>
<td>Address line 13</td>
<td>Address line 23</td>
</tr>
</tbody>
</table>

Month Date, Year

If instead of \and, we use (plain old) \\, the names are printed one below another.

We may leave some of these arguments empty; for example, the command \date{ } prints no date. Note, however, that if you simply omit the \date command itself, the current date will be printed. The command

\thanks{footnote text}

can be given at any point within the \title, \author or \date. It puts a marker at this point and places the footnote text as a footnote. (The general method of producing a footnote is to type \footnote{footnote text} at the point we want to refer to.)

As mentioned earlier, the “title” is printed in a separate page for the document classes book and report and in the first page of the document for the class article. (Also recall that this behavior can be modified by the options titlepage or notitlepage.)

\subsection{Abstract}

In the document classes article and report, an abstract of the document in special format can be produced by the commands

\begin{abstract}
Abstract Text
\end{abstract}

Note that we have to type the abstract ourselves. (There is a limit to what even \LaTeX{} can do.) In the report class this appears on the separate title page and in the article class it appears below the title information on the first page (unless overridden by the title page option). This command is not available in the book class.

\section{Dividing the Document}

A book is usually divided into chapters and (if it is technical one), chapters are divided into sections, sections into subsections and so on. \LaTeX{} provides the following hierarchy
of \textit{sectioning} commands in the \texttt{book} and \texttt{report} class:

\begin{verbatim}
\chapter
\section
\subsection
\subsubsection
\paragraph
\subparagraph
\end{verbatim}

Except for \texttt{chapter} all these are available in \texttt{article} class also. For example, the heading at the beginning of this chapter was produced by

\texttt{\chapter{The Document}}

and the heading of this section was produced by

\texttt{\section{Dividing the document}}

To see the other commands in action, suppose at this point of text I type

\texttt{\subsection{Example}}

In this example, we show how subsections and subsubsections are produced (there are no subsubsubsections). Note how the subsections are numbered.

\texttt{\subsubsection{Subexample}}

Did you note that subsubsections are not numbered? This is so in the \texttt{book} and \texttt{report} classes. In the \texttt{article} class they too have numbers. (Can you figure out why?)

\texttt{\paragraph{Note}}

Paragraphs and subparagraphs do not have numbers. And they have \texttt{run-in} headings.

Though named “\textit{paragraph}” we can have several paragraphs of text within this.

\texttt{\subparagraph{Subnote}}

Subparagraphs have an additional indentation too.

And they can also contain more than one paragraph of text.

We get

\subsection{Example}

In this example, we show how subsections and subsubsections are produced (there are no subsubsubsections). Note how the subsections are numbered.

\subexample

Did you note that subsubsections are not numbered? This is so in the \texttt{book} and \texttt{report} classes. In the \texttt{article} class they too have numbers. (Can you figure out why?)

\note Paragraphs and subparagraphs do not have numbers. And they have \texttt{run-in} headings. Though named “\textit{paragraph}” we can have several paragraphs of text within this.

\subnote Subparagraphs have an additional indentation too. And they can also contain more than one paragraph of text.
II.6.2. More on sectioning commands

In the book and the report classes, the \chapter command shifts to the beginning of a new page and prints the word “Chapter” and a number and beneath it, the name we have given in the argument of the command. The \section command produces two numbers (separated by a dot) indicating the chapter number and the section number followed by the name we have given. It does not produce any text like “Section”. Subsections have three numbers indicating the chapter, section and subsection. Subsubsections and commands below it in the hierarchy do not have any numbers.

In the article class, \section is highest in the hierarchy and produces single number like \chapter in book. (It does not produce any text like “Section”, though.) In this case, subsubsections also have numbers, but none below have numbers.

Each sectioning command also has a “starred” version which does not produce numbers. Thus \section*(name) has the same effect as \section(name), but produces no number for this section.

Some books and longish documents are divided into parts also. \part also has a \part command for such documents. In such cases, \part is the highest in the hierarchy, but it does not affect the numbering of the lesser sectioning commands.

You may have noted that \part has a specific format for typesetting the section headings, such as the font used, the positioning, the vertical space before and after the heading and so on. All these can be customized, but it requires some \TeXpertise and cannot be addressed at this point. However, the package sectsty provided some easy interfaces for tweaking some of these settings.

II.7. What next?

The task of learning to create a document in \TeX is far from over. There are other things to do such as producing a bibliography and a method to refer to it and also at the end of it all to produce a table of contents and perhaps an index. All these can be done efficiently (and painlessly) in \TeX, but they are matters for other chapters.
1 one inch + \hoffset 2 one inch + \voffset 3 \evensidemargin = 54pt 4 \topmargin = 18pt 5 \headheight = 12pt 6 \headsep = 18pt 7 \textheight = 609pt 8 \textwidth = 380pt 9 \marginparsep = 7pt 10 \marginparwidth = 115pt 11 \footskip = 25pt \hoffset = 0pt \marginparpush = 5pt (not shown) \offset = 0pt \paperwidth = 597pt \paperheight = 845pt
1 one inch + \hoffset
2 one inch + \voffset
3 \oddsidemargin = 18pt
4 \topmargin = 18pt
5 \headheight = 12pt
6 \headsep = 18pt
7 \textheight = 609pt
8 \textwidth = 380pt
9 \marginparsep = 7pt
10 \marginparwidth = 115pt
11 \footskip = 25pt
\hoffset = 0pt
\voffset = 0pt
\paperwidth = 597pt
\paperheight = 845pt
BIBLIOGRAPHY

III.1. INTRODUCTION

Bibliography is the environment which helps the author to cross-reference one publication from the list of sources at the end of the document. \LaTeX{} helps authors to write a well structured bibliography, because this is how \LaTeX{} works—by specifying structure.

It is easy to convert the style of bibliography to that of a publisher's requirement, without touching the code inside the bibliography. We can maintain a bibliographic database using the program Bib\TeX{}. While preparing the articles, we can extract the needed references in the required style from this data base. \harvard{} and \natbib{} are widely used packages for generating bibliography.

To produce bibliography, we have the environment \texttt{thebibliography}¹, which is similar to the \texttt{enumerate} environment. Here we use the command \texttt{\bibitem} to separate the entries in the bibliography and use \texttt{\cite} to refer to a specific entry from this list in the document. This means that at the place of citation, it will produce number or author-year code connected with the list of references at the end.

\begin{verbatim}
\begin{thebibliography}{widest-label}
  \bibitem{key1}
  \bibitem{key2}
\end{thebibliography}
\end{verbatim}

The \texttt{\begin{thebibliography}} command requires an argument that indicates the width of the widest label in the bibliography. If you know you would have between 10 and 99 citations, you should start with

\begin{verbatim}
\begin{thebibliography}{99}
\end{thebibliography}
\end{verbatim}

You can use any two digit number in the argument, since all numerals are of the same width. If you are using customized labels, put the longest label in argument, for example \texttt{\begin{thebibliography}}{Long-name}. Each entry in the environment should start with

\begin{verbatim}
\bibitem{key1}
\end{verbatim}

If the author name is Alex and year 1991, the key can be coded as ale\texttt{91} or some such mnemonic string². This \texttt{key} is used to cite the publication within the document text. To cite a publication from the bibliography in the text, use the \texttt{\cite} command, which takes with the corresponding key as the argument. However, the argument to \texttt{\cite} can also be two or more keys, separated by commas.

¹Bibliography environment need two compilations. In the first compilation it will generate file with \texttt{aux} extension, where \texttt{citation} and \texttt{bibcite} will be marked and in the second compilation \texttt{\cite} will be replaced by numeral or author-year code.

²Key can be any sequence of letters, digits and punctuation characters, except that it may not contain a comma (maximum 256 characters).
In bibliography, numbering of the entries is generated automatically. You may also add a note to your citation, such as page number, chapter number etc. by using an optional argument to the \cite command. Whatever text appears in this argument will be placed within square brackets, after the label.

\cite[key1]

See below an example of bibliography and citation. The following code

\begin{thebibliography}{9}
\bibitem{don89}Donald E. Knuth, 1989. \emph{Typesetting Concrete Mathematics}, TUGBoat, 10(1):31-36.
\bibitem{rondon89}Ronald L. Graham, Donald E. Knuth, and Ore Patashnik, 1989. \emph{Concrete Mathematics: A Foundation for Computer Science}, Addison-Wesley, Reading.
\end{thebibliography}

produces the following output:

\begin{itemize}
\item It is hard to write unstructured and disorganised documents using \LaTeX\cite{les85}. It is interesting to typeset one equation\cite[Sec 3.3]{les85} rather than setting ten pages of running matter\cite{don89,rondon89}.
\end{itemize}

The \natbib package is widely used for generating bibliography, because of its flexible interface for most of the available bibliographic styles. The \natbib package is a re-implementation of the \LaTeX \cite command, to work with both author-year and numerical citations. It is compatible with the standard bibliographic style files, such as plain.bst, as well as with those for harvard, apalike, chicago, astron, authordate, and of course \natbib. To load the package; use the command.

\usepackage[options]{natbib}
III.2.1. Options for natbib

round (default) for round parentheses
square for square brackets
curly for curly braces
angle for angle brackets
colon (default) to separate multiple citations with colons
comma to use commas as separators
authoryear (default) for author–year citations
numbers for numerical citations
super for superscripted numerical citations, as in Nature
sort orders multiple citations into the sequence in which they appear in the list of references
sort&compress as sort but in addition multiple numerical citations are compressed if possible (as 3–6, 15)
longnamesfirst makes the first citation of any reference the equivalent of the starred variant (full author list) and subsequent citations normal (abbreviated list)
sectionbib redefines \thebibliography to issue \section instead of \chapter; valid only for classes with a \chapter command; to be used with the chapterbib package
nonamebreak keeps all the authors’ names in a citation on one line; causes overfull hboxes but helps with some hyperref problems.

You can set references in the Nature style of citations (superscripts) as follows
\documentclass{article}
\usepackage{natbib}
\citestyle{nature}
\begin{document}
. . . . . .
. . . . . .
\end{document}

III.2.2. Basic commands

The natbib package has two basic citation commands, \citet and \citep for textual and parenthetical citations, respectively. There also exist the starred versions \citet* and \citep* that print the full author list, and not just the abbreviated one. All of these may take one or two optional arguments to add some text before and after the citation.

Normally we use author name and year for labeling the bibliography.
\begin{thebibliography}{widest-label}
\bibitem{Leslie(1985)}{les85}Leslie Lamport, 1985. \LaTeX---A Document Preparation...
\bibitem{Donale(00)}{don89}Donald E. Knuth, 1989. \Typesetting Concrete Mathematics,...
\bibitem{Ronald, Donald and Ore(1989)}{rondon89}Ronald L. Graham, ...
\end{thebibliography}

Year in parentheses is mandatory in optional argument for bibitem. If year is missing in any of the bibitem, the whole author–year citation will be changed to numerical citation. To avoid this, give ‘(0000)’ for year in optional argument and use partial citations (\citeauthor) in text.
Don't put 'space character' before opening bracket of year in optional argument.

\cite{ale91} ⇒ Alex et al. (1991)
\cite[chap. 4]{ale91} ⇒ Alex et al. (1991, chap. 4)
\citep{ale91} ⇒ (Alex et al., 1991)
\citep[chap. 4]{ale91} ⇒ (Alex et al., 1991, chap. 4)
\citep[see]{ale91} ⇒ (see Alex et al., 1991)
\citep[see, chap. 4]{jon91} ⇒ (see Alex et al., 1991, chap. 4)
\citep*[ale91] ⇒ Alex, Mathew, and Ravi (1991)
\citep*[ale91] ⇒ (Alex, Mathew, and Ravi, 1991)

### 3.2.5. Multiple citations

Multiple citations may be made as usual, by including more than one citation key in the \cite command argument.

\cite{ale91,rav92} ⇒ Alex et al. (1991); Ravi et al. (1992)
\citep{ale91,rav92} ⇒ (Alex et al., 1991; Ravi et al. 1992)
\citep{ale91,ale92} ⇒ (Alex et al., 1991, 1992)
\citep{ale91a,ale91b} ⇒ (Alex et al., 1991a, b)

### 3.2.4. Numerical mode

These examples are for author–year citation mode. In numerical mode, the results are different.

\cite{ale91} ⇒ Alex et al. [5]
\cite[chap. 4]{ale91} ⇒ Alex et al. [5, chap. 4]
\cite{ale91} ⇒ [5]
\citep{ale91} ⇒ [5, chap. 4]
\citep[see]{ale91} ⇒ [see 5]
\citep[see, chap. 4]{ale91} ⇒ [see 5, chap. 4]
\citep{ale91a,ale91b} ⇒ [5, 12]

### 3.2.5. Suppressed parentheses

As an alternative form of citation, \citealt is the same as \cite but without any parentheses. Similarly, \citealp is \citep with the parentheses turned off. Multiple references, notes, and the starred variants also exist.

\citealt{ale91} ⇒ Alex et al. 1991
\citealp{ale91} ⇒ Alex, Mathew, and Ravi 1991
\citealp{ale91} ⇒ Alex, 1991
\citealp*[ale91] ⇒ Alex, Mathew, and Ravi 1991
\citealp{ale91,ale92} ⇒ Alex et al., 1991; Alex et al., 1992
\citealp{pg. 7}{ale91} ⇒ Alex, 1991, pg. 7
\cite{short comm.} ⇒ (short comm.)

The \citeext command allows arbitrary text to be placed in the current citation parentheses. This may be used in combination with \citealp.
III.2.6. Partial citations

In author–year schemes, it is sometimes desirable to be able to refer to the authors without the year, or vice versa. This is provided with the extra commands

\citeauthor{ale91} ⇒ Alex et al.
\citeauthor*[ale91] ⇒ Alex, Mathew, and Ravi
\citeyear{ale91} ⇒ 1991
\citeyearpar{ale91} ⇒ (1991)

III.2.7. Citations aliasing

Sometimes one wants to refer to a reference with a special designation, rather than by the authors, i.e. as Paper I, Paper II. Such aliases can be defined and used, textually and/or parenthetically with:

\defcitealias{jon90}{Paper˜I}
\citetalias{ale91} ⇒ Paper I
\citepalias{ale91} ⇒ (Paper I)

These citation commands function much like \citet and \citep: they may take multiple keys in the argument, may contain notes, and are marked as hyperlinks.

III.2.8. Selecting citation style and punctuation

Use the command \bibpunct with one optional and six mandatory arguments:

1. The opening bracket symbol, default = (  
2. The closing bracket symbol, default = )  
3. The punctuation between multiple citations, default = ;  
4. The letter ‘n’ for numerical style, or ‘s’ for numerical superscript style, any other letter for author–year, default = author--year;  
5. The punctuation that comes between the author names and the year  
6. The punctuation that comes between years or numbers when common author lists are suppressed (default = ,);

The optional argument is the character preceding a post-note, default is a comma plus space. In redefining this character, one must include a space if that is what one wants.

Example 1

\bibpunct{[}{]}{,}{a}{,}{;}

changes the output of

\citep{jon90,jon91,jam92}

into

Example 2
\bibpunct[;]{()}{,}{a}{}{;}
changes the output of
\citep[and references therein]{jon90}
into
(Jones et al. 1990; and references therein).
Bibliographic database is a database in which all the useful bibliographic entries can be stored. The information about the various publications is stored in one or more files with the extension `.bib`. For each publication, there is a key that identifies it and which may be used in the text document to refer to it. And this is available for all documents with a list of reference in the field. This database is useful for the authors/researchers who are constantly referring to the same publications in most of their works. This database system is possible with the BibTeX program supplied with the LaTeX package.

### IV.1. The BibTeX Program

BibTeX is an auxiliary program to LaTeX that automatically constructs a bibliography for a LaTeX document from one or more databases. To use BibTeX, you must include in your LaTeX input file a `\bibliography` command whose argument specifies one or more files that contain the database. For example

```
\bibliography{database1,database2}
```

The above command specifies that the bibliographic entries are obtained from `database1.bib` and `database2.bib`. To use BibTeX, your LaTeX input file must contain a `\bibliographystyle` command. This command specifies the bibliography style, which determines the format of the source list. For example, the command

```
\bibliographystyle{plain}
```

specifies that entries should be formatted as specified by the `plain` bibliography style (`plain.bst`). We can put `\bibliographystyle` command anywhere in the document after the `\begin{document}` command.

### IV.2. BibTeX Style Files

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plain</td>
<td>Standard BibTeX style. Entries sorted alphabetically with numeric labels.</td>
</tr>
<tr>
<td>unsrt</td>
<td>Standard BibTeX style. Similar to plain, but entries are printed in order of citation, rather than sorted. Numeric labels are used.</td>
</tr>
<tr>
<td>alpha</td>
<td>Standard BibTeX style. Similar to plain, but the labels of the entries are formed from the author's name and the year of publication.</td>
</tr>
<tr>
<td>abbrv</td>
<td>Standard BibTeX style. Similar to plain, but entries are more compact, since first names, month, and journal names are abbreviated.</td>
</tr>
<tr>
<td>acm</td>
<td>Alternative BibTeX style, used for the journals of the Association for Computing Machinery. It has the author name (surname and first name) in small caps, and numbers as labels.</td>
</tr>
</tbody>
</table>
IV. Bibliographic Databases

**apa-like**  Alternative BibTeX style, used by the journals of the American Psychology Association. It should be used together with the BibTeX apalike package. The bibliography entries are formatted alphabetically, last name first, each entry having a hanging indentation and no label.

Examples of some other style files are:

```
abbrv.bst, abstract.bst, acm.bst, agsm.bst, alpha.bst, amsalpha.bst, authordate1.bst, authordate1-4.sty, bbs.bst, cbe.bst, cell.bst, dcu.bst, harvard.sty, ieeeetr.bst, jtb.bst, kluwer.bst, named.bst, named.sty, natbib.bst, natbib.bst, nature.sty, nature.bst, phcpc.bst, phiaeabst, phjcpc.bst, phrmp.bst plainyr.bst, siam.bst
```

Various organisations or individuals have developed style files that correspond to the house style of particular journals or editing houses. We can also customise a bibliography style, by making small changes to any of the .bst files, or else generate our own using the makebst program.

### IV.2.1. Steps for running BibTeX with BibTeX

1. Run \LaTeX, which generates a list of \cite references in its auxiliary file, .aux.
2. Run BibTeX, which reads the auxiliary file, looks up the references in a database (one or more .bib files, and then writes a file (the .bbl file) containing the formatted references according to the format specified in the style file (the .bst file). Warning and error messages are written to the log file (the .blg file). It should be noted that BibTeX never reads the original \LaTeX source file.
3. Run \LaTeX again, which now reads the .bbl reference file.
4. Run \LaTeX a third time, resolving all references.

Occasionally the bibliography is to include publications that were not referenced in the text. These may be added with the command

```
\nocite{key}
```

given anywhere within the main document. It produces no text at all but simply informs BibTeX that this reference is also to be put into the bibliography. With \nocite{*}, every entry in all the databases will be included, something that is useful when producing a list of all entries and their keys.

After running BibTeX to make up the .bbl file, it is necessary to process \LaTeX at least twice to establish both the bibliography and the in-text reference labels. The bibliography will be printed where the \bibliography command is issued; it in fact inputs the .bbl file.

### IV.3. Creating a Bibliographic Database

Though bibliographic database creation demands more work than typing up a list of references with the \thebibliography environment; it has a great advantage that, the entries need to be included in the database only once and are then available for all future publications even if a different bibliography style is demanded in later works, all the information is already on hand in the database for BibTeX to write a new \thebibliography environment in another format. Given below is a specimen of an entry in bibliographic database:

```latex
\BIBitem{knuth:86a,}
\TITLE{Author}="Donald E. Knuth",
```
The first word, prefixed @, determines the entry type. The entry type is followed by the reference information for that entry enclosed in curly braces {}. The very first entry is the key for the whole reference by which it is referred to in the \cite command. In the above example it is knuth:86a. The actual reference information is then entered in various fields, separated from one another by commas. Each field consists of a field name, an = sign, with optional spaces on either side, and the field text. The field names above are AUTHOR, TITLE, PUBLISHER, ADDRESS, and YEAR. The field text must be enclosed either in curly braces or in double quotation marks. However, if the text consists solely of a number, as for YEAR above, the braces or quotation marks may be left off.

For each entry type, certain fields are required, others are optional, and the rest are ignored. These are listed with the description of the various entry types below. If a required field is omitted, an error message will appear during the BibTeX run. Optional fields will have their information included in the bibliography if they are present, but they need not be there. Ignored fields are useful for including extra information in the database that will not be output, such as a comment or an abstract of a paper. Ignored fields might also be ones that are used by other database programs.

The general syntax for entries in the bibliographic database reads

```latex
@entry_type{key, 
   field_name = {field text},
   ...
   field_name = {field text} }
```

The names of the entry types as well as the field names may be written in capitals or lower case letters, or in a combination of both. Thus @BOOK, @book, and @bOOk are all acceptable variations.

The outermost pair of braces for the entire entry may be either curly braces {}, as illustrated, or parentheses (). In the latter case, the general syntax reads

```latex
@entry_type(key, ... ...) 
```

However, the field text may only be enclosed within curly braces [...] or double quotation marks ... as shown in the example above.

The following is a list of the standard entry types in alphabetical order, with a brief description of the types of works for which they are applicable, together with the required and optional fields that they take.

@article: Entry for an article from a journal or magazine.
required fields: author, title, journal, year.
optional fields: volume, number, pages, month, note.

@book: Entry for a book with a definite publisher.
required fields: author or editor, title, publisher, year.
optional fields: volume or number, series, address, edition, month, note.

@booklet: Entry for a printed and bound work without the name of a publisher or sponsoring organisation.
required fields: title.
optional fields: author, howpublished, address, month, year, note.
@conference: Entry for an article in conference proceedings.
required fields: author, title, booktitle, year.
optional fields: editor, volume or number, series, pages, address, month, organisation, publisher, note.

@inbook: Entry for a part (chapter, section, certain pages) of a book.
required fields: author or editor, title, chapter and/or pages, publisher, year.
optional fields: volume or number, series, type, address, edition, month, note.

@incolletion: Entry for part of a book that has its own title.
required fields: author, title, booktitle, publisher, year.
optional fields: editor, volume or number, series, type, chapter, pages, address, edition, month, note.

@inproceedings: Entry for an article in conference proceedings.
required fields: author, title, booktitle, year.
optional fields: editor, volume or number, series, pages, address, month, organisation, publisher, note.

@manual: Entry for technical documentation.
required fields: title.
optional fields: author, organisation, address, edition, month, year, note.

@masterthesis: Entry for a Master's thesis.
required fields: author, title, school, year.
optional fields: type, address, month, note.

@misc: Entry for a work that does not fit under any of the others.
required fields: none.
optional fields: author, title, howpublished, month, year, note.

@phdthesis: Entry for a PhD thesis.
required fields: author, title, school, year.
optional fields: type, address, month, note.

@proceedings: Entry for conference proceedings.
required fields: title, year.
optional fields: editor, volume or number, series, address, month, organisation, publisher, note.

@unpublished: Entry for an unpublished work with an author and title.
required fields: author, title, note.
optional fields: month, year.

IV.3.1. Example of a \TeX file (sample.tex) using bibliographical database (bsample.bib)
\begin{document}
\section{Example of Citations of Kind \texttt{plain}}
Citation of a normal book\cite{Eijkhout:1991} and an edited book\cite{Roth:postscript}. Now we cite an article written by a single author\cite{Felici:1991} and by multiple authors\cite{Mittlebatch/Schoepf:1990}. A reference to an article inside proceedings\cite{Yannis:1991}. We refer to a manual\cite{Dynatext} and a technical report\cite{Knuth:WEB}. A citation of an unpublished work\cite{EVH:Office}. A reference to a chapter in a book\cite{Wood:color} and to a PhD thesis\cite{Liang:1983}.\end{document}
An example of multiple citations:\cite{Eijkhout:1991,Roth:postscript}.

\bibliographystyle{plain} \% plain.bst
\bibliography{bsample} \% bsample.bib
\end{document}

IV.3.2. Procedure for producing references for the above file sample.tex which uses bibliographic data base bsample.bib

$latex sample % 1st run of LaTeX

$ bibtex sample % BibTeX run
% Then sample.bbl file will
% be produced

$latex sample % 2nd run of LaTeX

If still unresolved citation references

$latex sample % 3rd run of LaTeX
TABLE OF CONTENTS, INDEX AND GLOSSARY

V.1. Table of Contents

A table of contents is a special list which contains the section numbers and corresponding headings as given in the standard form of the sectioning commands, together with the page numbers on which they begin. Similar lists exist containing reference information about the floating elements in a document, namely, the list of tables and list of figures. The structure of these lists is simpler, since their contents, the captions of the floating elements, all are on the same level.

Standard \LaTeX{} can automatically create these three contents lists. By default, \LaTeX{} enters text generated by one of the arguments of the sectioning commands into the .toc file. Similarly, \LaTeX{} maintains two more files, one for the list of figures (.lof) and one for the list of tables (.lot), which contain the text specified as the argument of the \caption command for figures and tables.

\tableofcontents{} produces a table of contents. \listoffigures{} and \listoftables{} produce a list of figures and list of tables respectively. These lists are printed at the point where these commands are issued. Occasionally, you may find that you do not like the way \LaTeX{} prints a table of contents or a list of figures or tables. You can fine-tune an individual entry by using the optional arguments to the sectioning command or \caption command that generates it. Formatting commands can also be introduced with the \addtocontents{}. If all else fails, you can edit the .toc, .lof, .lot files yourself. Edit these files only when preparing the final version of your document, and use a \nofiles{} command to suppress the writing of new versions of the files.

V.1.1. Additional entries

The *-form sectioning commands are not entered automatically in the table of contents. \LaTeX{} offers two commands to insert such information directly into a contents file:

\addtocontents{}\{file\}\{text\} \addcontentsline{}\{file\}\{type\}\{text\}

file The extension of the contents file, usually toc, lof or lot.

\textbf{type} The type of the entry. For the toc file the type is normally the same as the heading according to the format of which an entry must be typeset. For the lof or lot files, figure or table is specified.

\textbf{text} The actual information to be written to the file mentioned. \LaTeX{} commands should be protected by \protect to delay expansion

The \addtocontents{} command does not contain a type parameter and is intended to enter user-specific formatting information. For example, if you want to generate additional spacing in the middle of a table of contents, the following command can be issued:

\addtocontents{toc}{\protect\vspace{2ex}}
The `\addcontentsline` instruction is usually invoked automatically by the document sectioning commands, or by the `\caption` commands. If the entry contains numbered text, then `\numberline` must be used to separate the section number (number) from the rest of the text for the entry (heading) in the text parameter:

\begin{verbatim}
\protect\numberline{number}{heading}
\end{verbatim}

For example, a `\caption` command inside a `figure` environment saves the text annotating the figure as follows:

\begin{verbatim}
\addcontentsline[lof][figure]{\protect\numberline{\thefigure}}{captioned text}
\end{verbatim}

Sometimes `\addcontentsline` is used in the source to complement the actions of standard \LaTeX. For instance, in the case of the starred form of the section commands, no information is written to the .toc file. So if you do not want a heading number (starred form) but an entry in the .toc file you can write something like:

\begin{verbatim}
\chapter*{Forward}
\addcontentsline{toc}{chapter}{\numberline{}Forward}
\end{verbatim}

This produces an indented “chapter” entry in the table of contents, leaving the space where the chapter number would go free. Omitting the `\numberline` command would typeset the word “Forward” flush left instead.

## V.1. Typesetting a contents list

As discussed above, contents lists consist of entries of different types, corresponding to the structural units that they represent. Apart from these standard entries, these lists may contain any commands. A standard entry is specified by the command:

\begin{verbatim}
\contentsline{type}{text}{page}
\end{verbatim}

- **type**: Type of the entry, e.g. section, or figure.
- **text**: Actual text as specified in the argument of the sectioning or `\caption` commands.
- **page**: Pagename.

Note that section numbers are entered as a parameter of the `\numberline` command to allow formatting with the proper indentation. It is also possible for the user to create a table of contents by hand with the help of the command `\contentsline`. For example:

\begin{verbatim}
\contentsline{section}{\numberline{2.4}Structure of the Table of Contents}{31}
\end{verbatim}

To format an entry in the table of contents files, standard \LaTeX makes use of the following command:

\begin{verbatim}
@dottedtocline{level}{indent}{numwidth}{text}{page}
\end{verbatim}

The last two parameters coincide with those of `\contentsline`, since the latter usually invokes `@dottedtocline` command. The other parameters are the following:

- **level**: The nesting level of an entry. This parameter allows the user to control how many nesting levels will be displayed. Levels greater than the value of counter `tocdepth` will not appear in the table of contents.
- **indent**: This is total indentation from the left margin.
- **numwidth**: The width of the box that contains the number if `text` has a `\numberline` command. This is also the amount of extra indentation added to the second and later lines of a multiple line entry.
Additionally, the command `\dottedtocline` uses the following formatting parameters, which specify the visual appearance of all entries:

- `\@pnumwidth` The width of the box in which the page number is set.
- `\@tocmarg` The indentation of the right margin for all but the last line of multiple line entries. Dimension, but changed with `\renewcommand`.
- `\@dotsep` The separation between dots, in mu (math units). It is a pure number (like 1.7 or 2). By making this number large enough you can get rid of the dots altogether. Changed with `\renewcommand` as well.

### V.1.3. Multiple tables of contents

The `minitoc` package, initially written by Nigel Ward and Dan Jurafsky and completely redesigned by Jean-Pierre Drucbert, creates a mini-table of contents (a “minitoc”) at the beginning of each chapter when you use the `book` or `report` classes.

The mini-table of contents will appear at the beginning of a chapter, after the `\chapter` command. The parameters that govern the use of this package are discussed below:

**Table V.1: Summary of the minitoc parameters**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\dominitoc</code></td>
<td>Must be put just in front of <code>\tableofcontents</code>, to initialize the minitoc system (Mandatory).</td>
</tr>
<tr>
<td><code>\faketableofcontents</code></td>
<td>This command replaces <code>\tableofcontents</code> when you want minitocs but not table of contents.</td>
</tr>
<tr>
<td><code>\minitoc</code></td>
<td>This command must be put right after each <code>\chapter</code> command where a minitoc is desired.</td>
</tr>
<tr>
<td><code>\minitocdepth</code></td>
<td>A \LaTeX{} counter that indicates how many levels of headings will be displayed in the minitoc (default value is 2).</td>
</tr>
<tr>
<td><code>\mtcindent</code></td>
<td>The length of the left/right indentation of the minitoc (default value is 24pt).</td>
</tr>
<tr>
<td><code>\mtcfont</code></td>
<td>Command defining the font that is used for the minitoc entries (The default definition is a small roman font).</td>
</tr>
</tbody>
</table>

For each mini-table, an auxiliary file with extension `.mtc<N>` where `<N>` is the chapter number, will be created.

By default, these mini-tables contain only references to sections and subsections. The `\minitocdepth` counter, similar to `\tocdepth`, allows the user to modify this behaviour.

As the minitoc takes up room on the first page(s) of a chapter, it will alter the page numbering. Therefore, three runs normally are needed to get correct information in the mini-table of contents.

To turn off the `\minitoc` commands, merely replace the package `minitoc` with `minitocoff` on your `\usepackage` command. This assures that all `\minitoc` commands will be ignored.

### V.2. INDEX

To find a topic of interest in a large document, book, or reference work, you usually turn to the table of contents or, more often, to the index. Therefore, an index is a very important part of a document, and most users’ entry point to a source of information is precisely through a pointer in the index. The most generally used index preparation program is `MakeIndex`.
Each \texttt{\index} command causes \LaTeX{} to write an entry in the .\texttt{idx} file. This command writes the text given as an argument, in the .\texttt{idx} file. This .\texttt{idx} will be generated only if we give \texttt{\makeindex} command in the preamble otherwise it will produce nothing.

\texttt{\index{index\_entry}}

To generate index follow the procedure given below:

1. Tag the words inside the document, which needs to come as index, as an argument of \texttt{\index} command.
2. Include the \texttt{makeidx} package with a \texttt{\usepackage} command and put \texttt{\makeindex} command at the preamble.
3. Put a \texttt{\printindex} command where the index is to appear, normally before \texttt{\end{document}} command.
4. \LaTeX{} file. Then a raw index (file.\texttt{idx}) will be generated.
5. Then run \texttt{makeindex}. (makeindex \texttt{file.idx} or \texttt{makeindex file}). Then two more files will be generated, \texttt{file.ind} which contains the index entries and \texttt{file.ilg}, a transcript file.
6. Then run \LaTeX{} again. Now you can see in the dvi that the index has been generated in a new page.

\subsection{Simple index entries}

Each \texttt{\index} command causes \LaTeX{} to write an entry in the .\texttt{idx} file. For example

\texttt{\index{animal\_entry}}
V.2. Sub entries

Up to three levels of index entries (main, sub, and subsub entries) are available with \textsc{LATEX-MakeIndex}. To produce such entries, the argument of the \texttt{index} command should contain both the main and subentries, separated by ! character.

Page 5: \texttt{\index{dimensions!rule!width}}

This will come out as

dimensions
rule
width, 5

V.2.3. Page ranges and cross-references

You can specify a page range by putting the command \texttt{\index{...|}} at the beginning of the range and \texttt{\index{...|)}} at the end of the range. Page ranges should span a homogeneous numbering scheme (e.g., Roman and Arabic page numbers cannot fall within the same range).

You can also generate cross-reference index entries without page numbers by using the \texttt{see} encapsulator. Since “see” entry does not print any page number, the commands \texttt{\index{...|see{...}}} can be placed anywhere in the input file after the \texttt{\begin{document}} command. For practical reasons, it is convenient to group all such cross-referencing commands in one place.

V.2.4. Controlling the presentation form

Sometimes you may want to sort an entry according to a key, while using a different visual representation for the typesetting, such as Greek letters, mathematical symbols, or specific typographic forms. This function is available with the syntax: \texttt{key@visual}, where \texttt{key} determines the alphabetical position and the string \texttt{value} produces the typeset text of the entry.

For some indexes certain page numbers should be formatted specially, with an italic page number (for example) indicating a primary reference, and an \texttt{n} after a page number denoting that the item appears in a footnote on that page. \textsc{MakeIndex} allows you to format an individual page number in any way you want by using the encapsulator syntax specified | character. What follows the | sign will “encapsulate” or enclose the page number associated with the index entry. For instance, the command \texttt{\index[keyword]{xxx|}} will produce a page number of the form \texttt{\xxx[n]}, where \texttt{n} is the page number in question.

Figure V.2: Page range and cross-referencing

<table>
<thead>
<tr>
<th>fonts</th>
<th>Page ii: \texttt{\index{table}}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Modern, 13–25</td>
<td>Page xi: \texttt{\index{table}}</td>
</tr>
<tr>
<td>math, \textit{see} math, fonts</td>
<td>Page 5: \texttt{\index{fonts!PostScript}}</td>
</tr>
<tr>
<td>PostScript, 5</td>
<td>Page 13: \texttt{\index{fonts!Computer Modern}}</td>
</tr>
<tr>
<td>table, ii–xi, 14</td>
<td>Page 14: \texttt{\index{table}}</td>
</tr>
<tr>
<td></td>
<td>Page 17: \texttt{\index{fonts!math</td>
</tr>
<tr>
<td></td>
<td>Page 21: \texttt{\index{fonts!Computer Modern}}</td>
</tr>
<tr>
<td></td>
<td>Page 25: \texttt{\index{fonts!Computer Modern}}</td>
</tr>
</tbody>
</table>
**delta, 14**

\( \delta, 23 \)

**delta wing, 16**

**flower, 19**

**ninety, 26**

**xc, 28**

**ninety-five, 5**

**tabbing, 7, 34–37**

**tabular, ii, 21, 22n**

**tabular environment, 23**

<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \delta )</td>
<td>23</td>
</tr>
<tr>
<td>( \text{tabular} ) environment</td>
<td>23</td>
</tr>
<tr>
<td>tabbing</td>
<td>7, 34–37</td>
</tr>
<tr>
<td>tabular</td>
<td>ii, 21, 22n</td>
</tr>
<tr>
<td>ninety</td>
<td>26</td>
</tr>
<tr>
<td>ninety-five</td>
<td>5</td>
</tr>
<tr>
<td>( \texttt{x} )</td>
<td>28</td>
</tr>
</tbody>
</table>

---

**Figure V.3: Controlling the presentation form**

\( \text{	exttt{\@} sign, 2} \)

<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \texttt{@} ) sign</td>
<td>2</td>
</tr>
</tbody>
</table>

---

**Figure V.4: Printing those special characters**

\( \text{	exttt{\@} see vertical bar} \)

\( \text{exclamation (!), 4} \)

\( \text{Ah!, 5} \)

\( \text{M"adchen, 3} \)

\( \text{quote ("), 1} \)

\( \text{" sign, 1} \)

<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \texttt{@} ) sign</td>
<td>2</td>
</tr>
<tr>
<td>( ! )</td>
<td>4</td>
</tr>
<tr>
<td>( \texttt{@} ) sign</td>
<td>5</td>
</tr>
<tr>
<td>( \texttt{M&quot;a} \texttt{dchen} )</td>
<td>3</td>
</tr>
<tr>
<td>( &quot; )</td>
<td>1</td>
</tr>
</tbody>
</table>

---

**V.2.5. Printing those special characters**

To typeset one of the characters having a special meaning to \textit{MakeIndex} (\( \!, \, \), \( \texttt{\@}, \text{or |} \)) in the index, precede it with a " character. More precisely, any character is said to be quoted if it follows an unquoted " that is not part of a " command. The latter case is for allowing umlaut characters. Quoted \( \!, \, \), \( \texttt{\@}, \text{or |} \) characters are treated like ordinary characters, losing their special meaning. The " preceding a quoted character is deleted before the entries are alphabetised.

---

**Glossary**

A 'glossary' is a special index of terms and phrases alphabetically ordered together with their explanations. To help set up a glossary, L\TeX offers the commands

\( \text{\texttt{\texttt{\texttt{\makeglossary}}} in the preamble and} \)

\( \text{\texttt{\texttt{\texttt{\glossary}}(glossary-entry)}} in the text part} \)
which function just like the commands for making up an index register. The entries are written to a file with extension .glo after the command \makeglossary has been given in the preamble. The form of these file entries from each \glossary command is
\glossaryentry\textit{glossary-entry}\{pagenumber\}

The information in the .glo file can be used to establish a glossary. However, there is no equivalent to the \theindex environment for a glossary, but a recommended structure is the description environment or a special list environment.
TUTORIAL VI

DISPLAYED TEXT

There are many instances in a document when we want to visually separate a portion of text from its surrounding material. One method of doing this is to typeset the distinguished text with added indentation. It is called *displaying*. \TeX{} has various constructs for displaying text depending on the nature of the displayed text.

VI.1. Borrowed Words

Quotations are often used in a document, either to add weight to our arguments by referring to a higher authority or because we find that we cannot improve on the way an idea has been expressed by someone else. If the quote is a one-liner, we can simply include it within double-quotes and be done with it (remember how to use quotes in \TeX{}?). But if the quotation is several lines long, it is better to display it. Look at the following example:

\begin{quote}
Some mathematicians elevate the spirit of Mathematics to a kind of intellectual aesthetics. It is best voiced by Bertrand Russell in the following lines.

The true spirit of delight, the exaltation, the sense of being more than man, which is the touchstone of the highest excellence, is to be found in Mathematics as surely as in poetry... Real life is, to most men, a long second best, a perpetual compromise between the ideal and the possible; but the world of pure reason knows no compromise, no practical limitations, no barriers to the creative activity embodying in splendid edifices the passionate aspiration after the perfect, from which all great work springs.

Yes, to men like Russell, Mathematics is more of an art than science.
\end{quote}

This was typeset as shown below:

\begin{verbatim}
Some mathematicians elevate the spirit of Mathematics to a kind of intellectual aesthetics. It is best voiced by Bertrand Russell in the following lines.
\begin{quote}
The true spirit of ..............................................................from which all great work springs.
\end{quote}
\end{verbatim}

Note that here we give instructions to \TeX{} to typeset some material in a separate paragraph with additional indentation on either side and indicate the start and end of material requiring special treatment, by means of the commands

```
\begin{quote} ... \end{quote}
```
This is an example of what is known in \LaTeX parlance as an \textit{environment}. Environments are used to delimit passages requiring special typographic treatments and to give instructions to \LaTeX on how to typeset it. The general form of an environment is of the form
\begin{center}
\textbf{VI. Displayed Text}
\end{center}
\begin{verbatim}
\begin{name} ... \end{name}
\end{verbatim}
where \texttt{name} is the name of the environment and signifies to \LaTeX the type of typographic treatment required (deliberate attempt at a pun, that).

The quoted part in this example is a single paragraph. If the quotation runs into several paragraphs, we must use the \texttt{quotation} environment, by enclosing the quotation within \texttt{\begin{quotation} and \end{quotation}}. As usual, paragraphs are separated by blank lines while typing the source file.

\textbf{VI.2. Poetry in typesetting}
\LaTeX can write poetry...well almost; if you write poems, \TeX can nicely typeset it for you. (I have also heard some \TeX wizards saying Knuth's code is sheer poetry!) Look at the passage below:

Contrary to popular belief, limericks are not always ribald. Some of them contain mathematical concepts:
\begin{verse}
A mathematician once confided
That a Mobius band is one sided
You'll get quite a laugh
If you cut it in half
For it stays in one piece when divided
\end{verse}
There is an extension of this to Klein's bottle also.

Note that line breaks are forced by the symbol `\'. Different stanzas are separated in the input by one (or more) blank lines. If you do not want \TeX to start a new page at a particular line break (if you want to keep rhyming couplets together in one page, for example), then use `\\*' instead of plain `\'. Again, if you want more space between lines than what \TeX deems fit, then use `\' with an optional \texttt{length} as in `\[5pt\] which adds an \textit{extra} vertical space of 5 points between the lines. You can also type `\[*5pt\]', whose intention should be obvious by now.

\textbf{VI.3. Making lists}
Lists are needed to keep some semblance of order in a chaotic world and \LaTeX helps us to typeset them nicely. Also, there are different kinds of lists available by default and if
none of them suits your need, there are facilities to tweak these or even design your own. Let us first have a look at the types of lists \( \LaTeX \) provides.

### VI.3.1. Saying it with bullets

The \texttt{itemize} environment gives us a bullet-list. For example it produces something like this:

One should keep the following in mind when using \LaTeX
\begin{itemize}
\item \LaTeX\ is a typesetting language and not a word processor
\item \LaTeX\ is a program and and not an application
\item There is no meaning in comparing \LaTeX\ to a word processor, since the design purposes are different
\end{itemize}

Being a program, \LaTeX\ offers a high degree of flexibility.

The input which produces this is given below:

\begin{verbatim}
One should keep the following in mind when using \LaTeX
\begin{itemize}
\item \LaTeX\ is a typesetting language and not a word processor
\item \LaTeX\ is a program and and not an application
\item There is no meaning in comparing \LaTeX\ to a word processor, since the design purposes are different
\end{itemize}
\end{verbatim}

We can have lists within lists. For example:

One should keep the following in mind when using \LaTeX
\begin{itemize}
\item \LaTeX\ is a typesetting language and not a word processor
\item \LaTeX\ is a program and and not an application
\item There is no meaning in comparing \LaTeX\ to a word processor, since the design purposes are different
\item \LaTeX\ is the natural choice in one of these situations
  \begin{itemize}
  \item If we want to typeset a document containing lot of Mathematics
  \item If we want our typed document to look beautiful
  \end{itemize}
\end{itemize}

Being a program, \LaTeX\ offers a high degree of flexibility.

It is produced by the input below:

\begin{verbatim}
One should keep the following in mind when using \LaTeX
\begin{itemize}
\item \LaTeX\ is a typesetting language and not a word processor
\item \LaTeX\ is a program and and not an application
\item There is no meaning in comparing \LaTeX\ to a word processor, since the design purposes are different
\item \LaTeX\ is the natural choice in one of these situations
  \begin{itemize}
  \item If we want to typeset a document containing lot of Mathematics
  \end{itemize}
\end{itemize}
\end{verbatim}
If we want our typed document to look beautiful
\end{itemize}
\end{itemize}
Being a program, \TeX \ offers a high degree of flexibility.

The \texttt{itemize} environment supports four levels of nesting. The full list of \texttt{labels} for the items (‘bullets’ for the first level, ‘dashes’ for the second and so on) is as shown below

- The first item in the first level
- the second item in the first level
  - The first item in the second level
  - the second item in the second level
    - The first item in the third level
    - the second item in the third level
      - The first item in the fourth level
      - the second item in the fourth level

Not satisfied with these default labels? How about this one?

\begin{itemize}
  \item First item of a new list
  \item Second item
\end{itemize}

It was produced by the following input:

\begin{verbatim}
{\renewcommand{\labelitemi}{$\triangleright$}
 \begin{itemize}
 \item First item of a new list
 \item Second item
 \end{itemize}}
\end{verbatim}

Several things need explanation here. First note that the first level labels of the \texttt{itemize} environment are produced by the (internal and so invisible to the user) command \texttt{\labelitemi} and by default, this is set as \texttt{\textbullet} to produce the default ‘bullets’. What we do here by issuing the \texttt{\renewcommand} is to override this by a choice of our own namely \$\triangleright\$ which produces the little triangles in the above list. Why the braces \{ and \} (did you notice them?) enclosing the whole input? They make the effect of the \texttt{\renewcommand} \textit{local} in the sense that this change of labels is only for this specific list. Which means the next time we use an \texttt{itemize} environment, the labels revert back to the original ‘bullets’. If we want the labels to be changed in the \textit{entire document}, then remove the braces.

What if we want to change the second level labels? No problem, just change the \texttt{\labelitemii} command, using a symbol of our choice. The third and fourth level labels are set by the commands (can you guess?) \texttt{\labelitemiii} and \texttt{\labelitemiv}. Look at the following example.
When the order of the items in a list is important, we need a list which specifies this order. For example, consider this

The three basic steps in producing a printed document using \TeX are as follows

1. Prepare a source file with the extension \texttt{tex}
2. Compile it with \TeX to produce a \texttt{dvi} file
3. Print the document using a \texttt{dvi} driver

Such a numbered list is produced by the \texttt{enumerate} environment in \TeX. The above list was produced by the following source.

\begin{enumerate}
\item prepare a source file with the extension "tex"
\end{enumerate}
\item Compile it with \LaTeX to produce a "dvi" file
\item Print the document using a "dvi" driver
\end{enumerate}

As in the case of \itemize environment, here also four levels of nesting are supports. The example below shows the labels used for different levels.

1. The first item in the first level
2. the second item in the first level
   (a) The first item in the second level
   (b) the second item in the second level
      i. The first item in the third level
      ii. the second item in the third level
         A. The first item in the fourth level
         B. the second item in the fourth level

How about customizing the labels? Here there is an additional complication in that the labels for items in the same level must follow a sequence (such as 1,2,3,... for the first level, (a), (b), (c),... for the second and so on, by default). There is a method for doing it, but it will take us into somewhat deeper waters. Fortunately, there is a package \enumerate by David Carlisle, which makes it easy. So if we want

\begin{enumerate}[i.]
\item Use a previewer (such as xdvi on \textsf{X Window System}) to view the output
\item Edit the source if needed
\item Recompile
\end{enumerate}

The three basic steps in producing a printed document using \TeX are as follows:

Step 1. Prepare a source file with the extension \textsf{tex}

Step 2. Compile it with \TeX to produce a \textsf{dvi} file
   \begin{enumerate}[i.]
   \item Use a previewer (such as xdvi on \textsf{X Window System}) to view the output
   \item Edit the source if needed
   \item Recompile
   \end{enumerate}

Step 3. Print the document using a \textsf{dvi} driver (such as dvips)

As you can see, the labels Step 1, Step 2 and Step 3 are produced by the \textit{optional} argument \texttt{Step 1} within square brackets immediately following the first \texttt{\begin{enumerate}} command and the labels i, ii, iii for the second level enumeration are produced by the optional \texttt{[i]} following the second \texttt{\begin{enumerate}}. So, what is \texttt{\hspace{0.5cm}} doing in the first optional argument? It is to provide an indentation at the left margin of the first level items, which the \enumerate environment does not produce by default.
We can add further embellishments. For example, if we want the labels in the first level of the above example to be in boldface, just change the optional argument \(\text{\texttt{\textbackslash hspace{0.5cm} Step 1}}\) to \(\text{\texttt{\textbackslash hspace{0.5cm}\textbfseries Step 1}}\). This produces:

The three basic steps in producing a printed document using \LaTeX\ are as follows:

**Step 1** Prepare a source file with the extension \texttt{tex}

**Step 2** Compile it with \LaTeX\ to produce a \texttt{dvi} file

(a) Use a previewer (such as \texttt{xdvi} on \texttt{X Window System}) to view the output
(b) Edit the source if needed
(c) Recompile

**Step 3** Print the document using a \texttt{dvi} driver (such as \texttt{dvips})

Some care must be taken when we give options like this. Suppose we want to produce something like this:

Addition of numbers satisfies the following conditions:

(A1) It is commutative
(A2) It is associative
(A3) There is an additive identity
(A4) Each number has an additive inverse

If we give the option \(\text{\texttt{\textbackslash hspace{1cm}(A1)}}\) as in

Addition of numbers satisfies the following conditions:

\begin{enumerate}
\item It is commutative
\item It is associative
\item There is an additive identity
\item Each number has an additive inverse
\end{enumerate}

Then we get the (somewhat surprising) output

\begin{enumerate}
\item It is commutative
\item It is associative
\item There is an additive identity
\item Each number has an additive inverse
\end{enumerate}

What happened? In the \texttt{enumerate} package, the option \texttt{[A]} signifies that we want the labels to be named in the sequence A, B, C,...,Z (the upper case Roman alphabet) and the option \texttt{[1]} signifies we want them as 1,2,3,... (the Arabic numerals). Other signifiers are \texttt{[a]} for lowercase Roman letters, \texttt{[I]} for uppercase Roman numerals and \texttt{[i]} for lowercase Roman numerals. So, if we use any one of these in the optional argument with some other purpose in mind, then \textit{enclose it in braces}. Thus the correct input to generate the above example is

Addition of numbers satisfies the following conditions

\begin{enumerate}
\item It is commutative
\item It is associative
\item There is an additive identity
\end{enumerate}
\item Each number has an additive inverse
\end{enumerate}

with braces surrounding the A. (The mystery is not over, is it? How come we got 11, 22, ... in the above example and not A1, B2, ...? Work it out yourselves!)

\section*{VI.5. Descriptions and Definitions}

There is a third type of list available off-the-shelf in \LaTeX which is used in typesetting lists like this

<table>
<thead>
<tr>
<th>Description</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\TeX</td>
<td>A typesetting program</td>
</tr>
<tr>
<td>Emacs</td>
<td>A text editor and also</td>
</tr>
<tr>
<td></td>
<td>a programming environment</td>
</tr>
<tr>
<td></td>
<td>a mailer</td>
</tr>
<tr>
<td></td>
<td>and a lot else besides</td>
</tr>
<tr>
<td>AbiWord</td>
<td>A word processor</td>
</tr>
</tbody>
</table>

This is produced by the description environment as shown below:

```
\begin{description}
\item[\TeX] A typesetting program
\item[Emacs] A text editor and also
  \begin{description}
  \item a programming environment
  \item a mailer
  \item and a lot else besides
  \end{description}
\item[AbiWord] A word processor
\end{description}
```

Note that this environment does not produce on its own any labels for the various items, but only produces as labels, whatever we give inside square brackets immediately after each item. By default, the labels are typeset in boldface roman. Also, there is no indentation for the first level. As with the other list environments, these can be changed to suit your taste. For example, suppose we want labels to be typeset in sans-serif roman and also want an indentation even for the first level. The code below will do the trick (remember why we include the whole input within braces?):

```
\renewcommand{\descriptionlabel}[1]{\hspace{1cm}\textsf{#1}}
\begin{description}
\item[\TeX] A typesetting program
\item[Emacs] A text editor and also
  \begin{description}
  \item a programming environment
  \item a mailer
  \item and a lot else besides
  \end{description}
\item[AbiWord] A word processor
\end{description}
```
and we get the output

```
Let us take stock of what we have learnt

\text{\LaTeX} \text{ A typesetting program}

\text{Emacs} \text{ A text editor and also}
\begin{itemize}
\item a programming environment
\item and a lot else besides
\end{itemize}

\text{AbiWord} \text{ A word processor}
```

Now is perhaps the time to talk about a general feature of all the three list environments we have seen. In any of these, we can override the default labels (if any) produced by the environment by something of our own by including it within square brackets immediately after the `\item`. Thus the input

```
The real number $l$ is the least upper bound of the
set $A$ if it satisfies the following conditions
\begin{enumerate}
\item[$(1)$] $l$ is an upper bound of $A$
\item[$(2)$] if $u$ is an upper bound of $A$, then $l \leq u$
\end{enumerate}
```

produces

```
The real number $l$ is the least upper bound of the set $A$ if it satisfies the following conditions
\begin{enumerate}
\item[$(1)$] $l$ is an upper bound of $A$
\item[$(2)$] if $u$ is an upper bound of $A$, then $l \leq u$
\end{enumerate}
```

The second condition is equivalent to

```
The second condition is equivalent to
\begin{enumerate}
\item[$(2)'$] If $a < l$, then $a$ is not an upper bound of $A$.
\end{enumerate}
```

This feature sometimes produces unexpected results. For example, if you type

```
Let’s review the notation
\begin{itemize}
\item $(0,1)$ is an \textit{open} interval
\item $[0,1]$ is a \textit{closed} interval
\end{itemize}
```

you will get

```
Let’s review the notation
\begin{itemize}
\item $(0,1)$ is an open interval
\item $[0,1]$ is a \textit{closed} interval
\end{itemize}
```

What happened? The $0,1$ within \textit{square brackets} in the second item is interpreted by \LaTeX as the \textit{optional label} for this item. The correct way to typeset this is
Let's review the notation
\begin{itemize}
\item $(0,1)$ is an \textbf{open} interval
\item $[0,1]$ is a \textbf{closed} interval
\end{itemize}

which produces

Let's review the notation
• $(0,1)$ is an \textit{open} interval
• $[0,1]$ is a \textit{closed} interval

So, why the dollars around $(0,1)$ also? Since $(0,1)$ and $[0,1]$ are \textit{mathematical entities}, the correct way to typeset them is to include them within braces in the input, even when there is no trouble such as with \texttt{item} as seen above. (By the way, do you notice any difference between $(0,1)$ produced by the input $(0,1)$ and $(0,1)$ produced by $(0,1)$?)

In addition to all these tweaks, there is also provision in \TeX\ to design your own 'custom' lists. But that is another story.
TUTORIAL VII
ROWS AND COLUMNS

The various list environments allows us to format some text into visually distinct rows. But sometimes the logical structure of the text may require these rows themselves to be divided into vertically aligned columns. For example, consider the material below typeset using the \texttt{description} environment (doesn’t it look familiar?)

\begin{tabbing}
Let’s take stock of what we’ve learnt\textbf{ Abiword} A word processor\textbf{ Emacs} A text editor\textbf{ \LaTeX} A typesetting program\end{tabbing}

A nicer way to typeset this is

\begin{tabbing}
Let’s take stock of what we’ve learnt\textbf{ AbiWord} A word processor\textbf{ Emacs} A text editor\textbf{ \LaTeX} A typesetting program\end{tabbing}

Here the three rows of text are visually separated into two columns of left aligned text. This was produced by the tabbing environment in \LaTeX.

VII.1. KEEPING TABS

VII.1.1. Basics

\begin{tabbing}
Let’s take stock of what we’ve learnt
\begin{verbatim}
\begin{tabbing}
\hspace{1cm}\= \textbf{AbiWord}\quad\= A word processor
\> \textbf{Emacs} \> A text editor
\> \textbf{\LaTeX} \> A typesetting program
\end{tabbing}
\end{verbatim}
\end{tabbing}

Let’s analyze it line by line. In the first line the first tab is put at a distance of 1 cm. from the left margin so that the text following it (‘AbiWord’ in boldface roman) starts from this point. The second tab is put at a distance of one \texttt{\quad} (this is an inbuilt length specification in \LaTeX roughly equal to one space) after the word ‘Abiword’ in boldface roman so that the text following it (‘A word processor’ in ordinary roman face) start from this point. The \texttt{\[5pt\]} command signifies the end of the first line and also asks for a vertical space of 5 points between the first and the second lines. In the second line,
the first ▷ command makes the text following it (‘Emacs’ in boldface roman) to start
from the first tab (already set in the first line), namely, 1 cm. from the left margin. The
second ▷ line makes the text following it (‘A text editor’ in ordinary roman face) at the
second tab already set, namely at a distance 1 cm plus the length of the word ‘AbiWord’
in boldface roman plus a ▷quad. The third line follows suit. The picture below will make
this clear.

One should be careful in setting tabs. For example to typeset

\begin{tabbing}
\textbf{\TeX} \quad \= A typesetting program\kill
\textbf{Emacs} \quad \> A text editor\kill
\textbf{AbiWord} \quad \> A word processor
\end{tabbing}

if you type
\begin{verbatim}
\begin{tabbing}
\textbf{\TeX} \quad \= A typesetting program\kill
\textbf{Emacs} \quad \> A text editor\kill
\textbf{AbiWord} \quad \> A word processor
\end{tabbing}
\end{verbatim}
then you end up with the output

\begin{tabbing}
\textbf{\TeX} \quad A typesetting program\kill
\textbf{Emacs} A text editor\kill
\textbf{AbiWord} A word processor
\end{tabbing}

Do you see what happened? The first line set the first tab (the only tab in this example) at
a distance of the length of the word ‘\TeX’ in boldface roman plus a ‘quad’ from the left
margin and the ▷ command in the second line makes the text following to start from
this tab, which is right next to the word ‘Emacs’ in this line. the same thing happens
in the third line, which is worse, since the position of the tab is at the ‘o’ of ‘AbiWord’
and the next word ‘A word processor’ starts from this point, and overwrites the previous
word. The correct way to obtain the output we want is to use a dummy line to mark the
tabs, without actually typesetting that line. This is achieved by the \kill command in
the tabbing environment, as shown below
\begin{verbatim}
\begin{tabbing}
\textbf{AbiWord} \quad \= A word processor\kill
\textbf{\TeX} \quad \> A typesetting program
\end{tabbing}
\end{verbatim}
VII.1. Keeping tabs

\textbf{Emacs} \quad A text editor
\textbf{AbiWord} \quad A word processor

New tabs, in addition to the ones already set by the first line (dummy or otherwise), can be set in any subsequent line. Thus the output

| \textbf{\TeX}   | : A typesetting program |
| \textbf{Emacs}  | : A text editor          |
| \textbf{AbiWord}| : A word processor       |

\begin{tabbing}
\textbf{AbiWord} \= : A word processor\kill
\textbf{\TeX} \> : A typesetting program\[[5pt]
\textbf{Emacs} \> : A text editor\[[5pt]
\> \> \quad \= : a programming environment\[[5pt]
\> \> \> \quad : a mail reader\[[5pt]
\> \> \> \> \quad : and a lot more besides\[[5pt]
\end{tabbing}

Here the first line sets two tabs and the fourth line sets a third tab after these two. All the three tabs can then be used in the subsequent lines. New tab positions which change the ones set up by the first line, can also be introduced in any line by the \= command. Thus we can produce

| Program | \textbf{\TeX} |
| Author  | Donald Knuth  |
| Manuals |

\begin{tabbing}
\textsf{Title} \textsf{Author} \textsf{Publisher}\kill
\textbf{The \TeX Book} Donald Knuth Addison-Wesley
\textbf{The Advanced \TeX Book} David Salomon Springer-Verlag\kill
\>\textsf{Program} \quad \= : \= \textbf{\TeX }\[[5pt]
\>\textsf{Author} \quad \> : \> Donald Knuth\[[5pt]
\>\textsf{Manuals} \quad \> :::\kill
\> \quad := \textbf{The Advanced }\textbf{\TeX Book} \quad \= := Springer-Verlag\kill
\>\textsf{\TeX Book} \quad \textsf{\TeX Book} \quad \textsf{David Salomon} \quad \textsf{Springer-Verlag} \[[8pt]
VII.1.2. Pushing and popping

What if you change the tab positions and then want the original settings back? Here’s where the command pair \pushtabs ... \poptabs is useful. Thus to typeset

```
\begin{tabbing}
Program : \TeX\quad Author : Donald Knuth
Manuals :
\pushtabs
\quad Title Author Publisher
\quad The \TeX\ Book Donald Knuth Addison-Wesley
\quad The Advanced \TeX\ Book David Salomon Springer-Verlag
Tutorial : http://tug.org.in/tutorial
\poptabs
\end{tabbing}
```

we type

```
\begin{tabbing}
Program\quad \= : \= \TeX\[5pt]
Author \> : \> Donald Knuth[5pt]
Manuals \> : \>
\pushtabs
\quad \= The Advanced \TeX\ Book \quad \= David Salomon \quad
\= Springer-Verlag\kill
\> Title Author Publisher
\> The \TeX\ Book Donald Knuth Addison-Wesley
\> The Advanced \TeX\ Book David Salomon Springer-Verlag
\poptabs
Tutorial \> : \>
"http://tug.org.in/tutorial"
\end{tabbing}
```

Here the first three lines follow a tabbing scheme, the next three lines follow another tabbing scheme and the last line reverts back to the original scheme. Here the \pushtabs command stores the current tabbing scheme and removes it so that a new tabbing scheme can be set up; and the \poptabs commands reactivates the original scheme. These commands can be nested.

VII.1.3. More commands

There are some more useful commands available in the tabbing environment. The \+ command given at the end of a line makes every subsequent line start at the first tab; with \+-+ at the end of a line, all subsequent lines start at the second tab and so on. The effect of each \+ can be neutralized by one \- command at the end of a line. The
command \(< at the beginning of a line neutralizes the effect of one \( command for that particular line.

The command \(' (left quote) puts the text following flush right against the right margin. Naturally we cannot use a \(' or \(' after this in a line.

Another interesting command is \(' (right quote). Within the tabbing environment an input of the form left_text \(' right_text puts the right_text at the current tab and the left_text just before this tab with a bit of spacing (preassigned by the parameter \tabbingsep).

The example below illustrates all the tabbing commands we’ve discussed

\begin{tabbing}
Row 1 Column 1\hspace{2cm} \\( Row 1 Column 2\hspace{5pt}
\> Row 2 Column 2\hspace{1.5cm} \( Row 2 Column 3\hspace{5pt}
Row 3 Column 3\hspace{5pt}
Row 4 Column 2 \> Row 4 Column 3\hspace{5pt}
\< Row 5 Column 1 \> Row 5 Column 2 \> Row 5 Column 3\hspace{5pt}
\> Row 6 Column 2 \> Row 6 Column 3\hspace{5pt}
Row 7 Column 1 \> Row 7 Column 2 \> Row 7 Column 3\hspace{5pt}
Row 8 Column 1 \> \quad \> Right\hspace{5pt}
Row 9 Column 1 \> and\> Row 9 Column 2\hspace{5pt}\poptabs
\quad \> Row 10 New Column 1\hspace{2.5cm} \> Row 10 New Column 2\hspace{5pt}
\> Row 11 New Column 2 \> Row 11 New Column 2\hspace{5pt}\poptabs
\quad \> Row 12 Old Column 1 \> Row 12 Old Column 2 \> Row 12 Old Column 3\end{tabbing}

It produces the following output

<table>
<thead>
<tr>
<th>Row 1 Column 1</th>
<th>Row 1 Column 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row 2 Column 2</td>
<td>Row 2 Column 3</td>
</tr>
<tr>
<td>Row 3 Column 3</td>
<td></td>
</tr>
<tr>
<td>Row 4 Column 2</td>
<td>Row 4 Column 3</td>
</tr>
<tr>
<td>Row 5 Column 1</td>
<td>Row 5 Column 2</td>
</tr>
<tr>
<td>Row 5 Column 3</td>
<td></td>
</tr>
<tr>
<td>Row 6 Column 2</td>
<td>Row 6 Column 3</td>
</tr>
<tr>
<td>Row 7 Column 1</td>
<td>Row 7 Column 2</td>
</tr>
<tr>
<td>Row 7 Column 3</td>
<td></td>
</tr>
<tr>
<td>Row 8 Column 1</td>
<td>and Row 9 Column 2</td>
</tr>
<tr>
<td>Row 9 Column 1</td>
<td>Row 9 Column 2</td>
</tr>
<tr>
<td>Row 10 New Column 1</td>
<td>Row 10 New Column 2</td>
</tr>
<tr>
<td>Row 11 New Column 2</td>
<td>Row 11 New Column 2</td>
</tr>
<tr>
<td>Row 12 Old Column 1</td>
<td>Row 12 Old Column 2</td>
</tr>
<tr>
<td>Row 12 Old Column 3</td>
<td></td>
</tr>
</tbody>
</table>

Recall that the commands \(, \(' and \(' are used for various accents outside the tabbing environment. If these are needed within the tabbing environment, they can be produced with the commands \a=, \a' or \a' commands.

One final word. You might’ve noted in the examples above that we give a sort of
'formatting' to the sources also. This is not really necessary from the point of view of \LaTeX since the output of the last example is the same even if we input

\begin{tabbing}
Row 1 Column 1 \hspace{2cm} Row 1 Column 2 \[5pt]\[5pt]
Row 2 Column 1 \hspace{1.5cm} Row 2 Column 2 \[5pt]\[5pt]
Row 3 Column 3 \[5pt]\[5pt]
Row 4 Column 2 \> Row 4 Column 3 \[5pt]\[5pt]
Row 5 Column 2 \> Row 5 Column 3 \[5pt]\[5pt]
Row 6 Column 2 \> Row 6 Column 3 \[5pt]\[5pt]
Row 7 Column 1 \> Row 7 Column 2 \> Row 7 Column 3 \[5pt]\[5pt]
Row 8 Column 1 \> \textbf{Flush right} \[5pt]\[5pt]
Row 9 Column 1 \> and' Row 9 Column 2 \[5pt]\[5pt]
\textbf{Flush right} \[5pt]\[5pt]
Row 10 New Column 1 \hspace{2.5cm} Row 10 New Column 2 \[5pt]\[5pt]
Row 11 New Column 2 \> Row 11 New Column 2 \[5pt]\[5pt]
\textbf{Flush right} \[5pt]\[5pt]
Row 12 Old Column 1 \> Row 12 Old Column 2 \> Row 12 Old Column 3 \[5pt]\[5pt]
\textbf{Flush right} \[5pt]\[5pt]
\textbf{Flush right} \[5pt]\[5pt]
\end{tabbing}

\LaTeX can make sense out of this, but we humans cannot. And such a jumble makes editing a hopeless task. The moral? Keep the source (humanly) readable.

\section{Tables}

Another way to format text into columns and rows is to use the tabular environment. Let's see it in action by means of an example.

<table>
<thead>
<tr>
<th>Planet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercury</td>
</tr>
<tr>
<td>Venus</td>
</tr>
<tr>
<td>Earth</td>
</tr>
<tr>
<td>Mars</td>
</tr>
<tr>
<td>Jupiter</td>
</tr>
<tr>
<td>Saturn</td>
</tr>
<tr>
<td>Uranus</td>
</tr>
<tr>
<td>Neptune</td>
</tr>
<tr>
<td>Pluto</td>
</tr>
</tbody>
</table>

\begin{center}
\begin{tabular}{lr}
Mercury & 4878 \\
Venus & 12104 \\
Earth & 12756 \\
Mars & 6794 \\
Jupiter & 142984 \\
Saturn & 120536 \\
Uranus & 51118 \\
Neptune & 49532 \\
Pluto & 2274 \\
\end{tabular}
\end{center}

As can be seen, Pluto is the smallest and Jupiter the largest.
As can be seen, Pluto is the smallest and Jupiter the largest.

The \begin{center} ... \end{center} commands centralize the table. The table itself is produced by the \begin{tabular} ... \end{tabular} commands. The \{lr\} specification immediately after the \begin{tabular} indicates there are two columns in the table with the entries in the first column aligned on the left and the entries in the second column aligned on the right. The entries in each column are separated by the & symbol and the termination of each row is signalled by the \ symbol. The \[5pt\] after the first row specifies as usual, an additional vertical space of 5 points after this row in the output.

In addition to the column specifiers l and r we also have a specifier c which makes the entries in the corresponding column centrally aligned. For example the input

\begin{center}
\begin{tabular}{cr}
    Planet & Diameter(km) \\
    Mercury & 4878 \\
    Venus & 12104 \\
    Earth & 12756 \\
    Mars & 6794 \\
    Jupiter & 142984 \\
    Saturn & 120536 \\
    Uranus & 51118 \\
    Neptune & 49532 \\
    Pluto & 2274 \\
\end{tabular}
\end{center}

produces the output below

<table>
<thead>
<tr>
<th>Planet</th>
<th>Diameter(km)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercury</td>
<td>4878</td>
</tr>
<tr>
<td>Venus</td>
<td>12104</td>
</tr>
<tr>
<td>Earth</td>
<td>12756</td>
</tr>
<tr>
<td>Mars</td>
<td>6794</td>
</tr>
<tr>
<td>Jupiter</td>
<td>142984</td>
</tr>
<tr>
<td>Saturn</td>
<td>120536</td>
</tr>
<tr>
<td>Uranus</td>
<td>51118</td>
</tr>
<tr>
<td>Neptune</td>
<td>49532</td>
</tr>
<tr>
<td>Pluto</td>
<td>2274</td>
</tr>
</tbody>
</table>

There’s yet another column specifier p which allows us to set column entries in a box of specified width (technically a “parbox”—see Chapter X). Suppose you want something like this
<table>
<thead>
<tr>
<th>Planet</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercury</td>
<td>Lunar like crust, crustal faulting, small magnetic fields.</td>
</tr>
<tr>
<td>Venus</td>
<td>Shrouded in clouds, undulating surface with highlands, plains, lowlands and craters.</td>
</tr>
<tr>
<td>Earth</td>
<td>Oceans of water filling lowlands between continents, unique in supporting life, magnetic field.</td>
</tr>
<tr>
<td>Mars</td>
<td>Cratered uplands, lowland plains, volcanic regions.</td>
</tr>
<tr>
<td>Jupiter</td>
<td>Covered by clouds, dark ring of dust, magnetic field.</td>
</tr>
<tr>
<td>Saturn</td>
<td>Several cloud layers, magnetic field, thousands of rings.</td>
</tr>
<tr>
<td>Uranus</td>
<td>Layers of cloud and mist, magnetic field, some rings.</td>
</tr>
<tr>
<td>Neptune</td>
<td>Unable to detect from earth.</td>
</tr>
<tr>
<td>Pluto</td>
<td>Unable to detect from earth.</td>
</tr>
</tbody>
</table>

It is produced from the input

\begin{center}
\begin{tabular}{lp{6cm}}
  Planet & Features \\
  Mercury & Lunar like crust, crustal faulting, small magnetic fields. \\
  Venus & Shrouded in clouds, undulating surface with highlands, plains, lowlands and craters. \\
  Earth & Oceans of water filling lowlands between continents, unique in supporting life, magnetic field. \\
  Mars & Cratered uplands, lowland plains, volcanic regions. \\
  Jupiter & Covered by clouds, dark ring of dust, magnetic field. \\
  Saturn & Several cloud layers, magnetic field, thousands of rings. \\
  Uranus & Layers of cloud and mist, magnetic field, some rings. \\
  Neptune & Unable to detect from earth. \\
  Pluto & Unable to detect from earth \\
\end{tabular}
\end{center}

Here the specification p(6cm) shows that in a “paragraph box” of width 6 cm. In a p-type column, if a \raggedright or \centering is given, then we can induce explicit line breaks within that column by the \ command. If such commands are used in the last column of a row, then the command \textbackslash{tabularnewline} should be used to terminate that row as in this example:

\begin{center}
\begin{tabular}{lp{6cm}}
  Planet & Features \\
  Mercury & \textbackslash{tabularnewline}Lunar like crust \\
  & \textbackslash{tabularnewline}Crustal faulting \\
  & \textbackslash{tabularnewline}Small magnetic fields \\
  Venus & \textbackslash{tabularnewline}Shrouded in clouds \\
  & \textbackslash{tabularnewline}Undulating surface \\
  Earth & \textbackslash{tabularnewline}Oceans of water \\
  & \textbackslash{tabularnewline}Unique in supporting life \\
  & \textbackslash{tabularnewline}Magnetic field \\
  Mars & \textbackslash{tabularnewline}Cratered uplands \\
\end{tabular}
\end{center}
VII.2. Tables

<table>
<thead>
<tr>
<th>Planet</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mercury</td>
<td>Lunar like crust</td>
</tr>
<tr>
<td></td>
<td>Crustal faulting</td>
</tr>
<tr>
<td></td>
<td>Small magnetic field</td>
</tr>
<tr>
<td>Venus</td>
<td>Shrouded in clouds</td>
</tr>
<tr>
<td></td>
<td>Undulating surface</td>
</tr>
<tr>
<td>Earth</td>
<td>Oceans of water</td>
</tr>
<tr>
<td></td>
<td>Unique in supporting life</td>
</tr>
<tr>
<td></td>
<td>Magnetic field</td>
</tr>
<tr>
<td>Mars</td>
<td>Cratered uplands</td>
</tr>
<tr>
<td></td>
<td>Lowland plains</td>
</tr>
<tr>
<td></td>
<td>Volcanic regions</td>
</tr>
<tr>
<td>Jupiter</td>
<td>Covered by clouds</td>
</tr>
<tr>
<td></td>
<td>Dark ring of dust</td>
</tr>
<tr>
<td></td>
<td>Magnetic field</td>
</tr>
<tr>
<td>Saturn</td>
<td>Several cloud layers</td>
</tr>
<tr>
<td></td>
<td>Magnetic field</td>
</tr>
<tr>
<td></td>
<td>Thousands of rings</td>
</tr>
<tr>
<td>Uranus</td>
<td>Layers of cloud and mist</td>
</tr>
<tr>
<td></td>
<td>Magnetic field</td>
</tr>
<tr>
<td></td>
<td>Some rings</td>
</tr>
<tr>
<td>Neptune</td>
<td>Unable to detect</td>
</tr>
<tr>
<td></td>
<td>from earth</td>
</tr>
<tr>
<td>Pluto</td>
<td>Unable to detect</td>
</tr>
<tr>
<td></td>
<td>from earth</td>
</tr>
</tbody>
</table>

This produces the output below

Note that the last two lines don’t need a `\raggedright` command, since there are no explicit linebreaks in them.

A table usually contains horizontal and vertical lines separating the rows and columns. These can also be produced in the `tabular` environment. For example, the first table we saw above can be typeset as
by the input
\begin{center}
\begin{tabular}{|l|r|}
\hline
Planet & Diameter(km) \\
\hline
Mercury & 4878 \\
Venus & 12104 \\
Earth & 12756 \\
Mars & 6794 \\
Jupiter & 142984 \\
Saturn & 120536 \\
Uranus & 51118 \\
Neptune & 49532 \\
Pluto & 2274 \\
\hline
\end{tabular}
\end{center}

Do you see what produced the vertical and horizontal lines? Instead of the specification
\{lr\} used earlier, we now have \{||r\}. The character | causes a vertical line to be drawn
at the specified location, running down the entire height of the table. (Two |'s in succes-
sion produce a double vertical lines.) An \hline command after a row draws a horizontal
line after that row, running along the entire width of the table. (Again, two \hline's in
succession produce a double horizontal line.) Note also that because of the last \hline,
we should give a line termination command \\ at the end of the last row also.

Now suppose we want to produce something like this

\begin{center}
\begin{tabular}{l|rr}
Planet & \multicolumn{2}{c|}{Distance from the sun (km)} \\
& Maximum & Minimum \\
Mercury & 69400000 & 46800000 \\
Venus & 109000000 & 107600000 \\
Earth & 152600000 & 147400000 \\
Mars & 249200000 & 207300000 \\
Jupiter & 817400000 & 741600000 \\
Saturn & 3011000000 & 2740000000 \\
Uranus & 3011000000 & 2740000000 \\
\hline
\end{tabular}
\end{center}

Here, there are three columns and the entry Distance from the sun (km) is to span the
the last two columns below it. The command \multicolumn does the trick as shown
below

\begin{center}
\begin{tabular}{|l|rr}
Planet & \multicolumn{2}{c|}{Distance from sun (km)} \\
& & \\
& Maximum & Minimum \\
\end{tabular}
\end{center}
Mercury & 69400000 & 46800000 \\
Venus & 10900000 & 107600000 \\
Earth & 152600000 & 147400000 \\
Mars & 249200000 & 207300000 \\
Jupiter & 817400000 & 741600000 \\
Saturn & 1512000000 & 1346000000 \\
Uranus & 3011000000 & 2740000000 \\
\end{tabular}

The entry \texttt{\multicolumn{2}{c}{Distance from sun (km)}} indicates that the item within the last set of braces is to span two columns as specified by the 2 within the first set of braces. The entry \texttt{c} within the second set of braces indicates that this text is to be centered within the column. Thus the general form of the command is

\texttt{\multicolumn{num}{pos}{item}}

where \texttt{num} is the number of columns to be spanned, \texttt{pos} is the position of the item within the column and \texttt{item} is the text of the item. Note also that the input for the second row starts with an \texttt{&} character. This is because there is no entry in the first column of the second row.

Now what if you want

<table>
<thead>
<tr>
<th>Planet</th>
<th>Distance from sun (km)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum</td>
</tr>
<tr>
<td>Mercury</td>
<td>69400000</td>
</tr>
<tr>
<td>Venus</td>
<td>10900000</td>
</tr>
<tr>
<td>Earth</td>
<td>152600000</td>
</tr>
<tr>
<td>Mars</td>
<td>249200000</td>
</tr>
<tr>
<td>Jupiter</td>
<td>817400000</td>
</tr>
<tr>
<td>Saturn</td>
<td>1512000000</td>
</tr>
<tr>
<td>Uranus</td>
<td>3011000000</td>
</tr>
<tr>
<td>Neptune</td>
<td>4541000000</td>
</tr>
<tr>
<td>Pluto</td>
<td>7346000000</td>
</tr>
</tbody>
</table>

Here the first few lines and the last lines of the input are as below (the other lines are the same as in the previous example).

\begin{center}
\begin{tabular}{|l|l|}
\hline
Planet & \texttt{\multicolumn{2}{c}{Distance from sun (km)}} \\
\cline{2-3}
& Maximum & Minimum \\
\hline

\texttt{\end{tabular}}
\end{center}

Note that the position specifier in the \texttt{\multicolumn} command here is \texttt{c}. This has to do with the way the environment splits the column specification into various columns.
For example, the specification \[|l|r|r|\] in this example is split into \[|l|, r| and r|\] and the \texttt{\multicolumn{2}} command resets the last two columns. In particular, the final \texttt{|} gets reset and we'll have to explicitly supply it in the position specification of the \texttt{\multicolumn} command as \texttt{c|}.

Note also the command \texttt{\cline{2-3}} after the first row. This draws a horizontal line from the second to the third column. In general the command \texttt{\cline{\textit{i-j}}} draws a horizontal line from the \texttt{i}th column to the \texttt{j}th column.

Another feature of the \texttt{\multicolumn} command is that with \texttt{\multicolumn{1}} we can override the position specification of any column set at the beginning of the environment. For example, consider the input below.

\begin{verbatim}
\begin{center}
\begin{tabular}{|l|r|r|}
\hline
\multicolumn{2}{|p{3.5cm}|}{\centering Distance from sun (million km)}\\\
\cline{2-3}
\multicolumn{1}{|c|}{Planet} & \multicolumn{1}{|c|}{Maximum} & \multicolumn{1}{|c|}{Minimum}\\
\hline
Mercury & 69.4 & 46.8\\
Venus & 109.0 & 107.6\\
Earth & 152.6 & 147.4\\
Mars & 249.2 & 207.3\\
Jupiter & 817.4 & 741.6\\
Saturn & 1512.0 & 1346.0\\
Uranus & 3011.0 & 2740.0\\
Neptune & 4543.0 & 4466.0\\
Pluto & 7346.0 & 4461.0\\
\hline
\end{tabular}
\end{center}
\end{verbatim}

It produces the output below

<table>
<thead>
<tr>
<th>Planet</th>
<th>Distance from sun (million km)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum</td>
</tr>
<tr>
<td>Mercury</td>
<td>69.4</td>
</tr>
<tr>
<td>Venus</td>
<td>109.0</td>
</tr>
<tr>
<td>Earth</td>
<td>152.6</td>
</tr>
<tr>
<td>Mars</td>
<td>249.2</td>
</tr>
<tr>
<td>Jupiter</td>
<td>817.4</td>
</tr>
<tr>
<td>Saturn</td>
<td>1512.0</td>
</tr>
<tr>
<td>Uranus</td>
<td>3011.0</td>
</tr>
<tr>
<td>Neptune</td>
<td>4543.0</td>
</tr>
<tr>
<td>Pluto</td>
<td>7346.0</td>
</tr>
</tbody>
</table>

Note that even though \texttt{\centering} is used in the last column of the first row, no \texttt{\tabularnewline} is required to terminate this row, since the scope of the \texttt{\centering} is limited by the \texttt{\multicolumn}.
By the way, do you feel that the tables we’ve been produced look a bit cramped? A bit crowded vertically? Well, you can create a bit more room between rows by redefining the value of \textbackslash arraystretch. By default, it’s value is 1 and if you set it to a number \( k \), then the interrow space is increased \( k \)-fold. Thus the input of the last example with the command

\begin{verbatim}
\texttt{\renewcommand{\arraystretch}{1.2}}
\end{verbatim}

after the \texttt{\begin{center}} produces

<table>
<thead>
<tr>
<th>Planet</th>
<th>Distance from sun (million km)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum</td>
</tr>
<tr>
<td>Mercury</td>
<td>69.4</td>
</tr>
<tr>
<td>Venus</td>
<td>109.0</td>
</tr>
<tr>
<td>Earth</td>
<td>152.6</td>
</tr>
<tr>
<td>Mars</td>
<td>249.2</td>
</tr>
<tr>
<td>Jupiter</td>
<td>817.4</td>
</tr>
<tr>
<td>Saturn</td>
<td>1512.0</td>
</tr>
<tr>
<td>Uranus</td>
<td>3011.0</td>
</tr>
<tr>
<td>Neptune</td>
<td>4543.0</td>
</tr>
<tr>
<td>Pluto</td>
<td>7346.0</td>
</tr>
</tbody>
</table>

Next let’s see how we produce a table like the one below

<table>
<thead>
<tr>
<th>Height (cm)</th>
<th>Ideal weight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>155</td>
<td>53.5–64</td>
</tr>
<tr>
<td>160</td>
<td>56–67</td>
</tr>
<tr>
<td>165</td>
<td>59–71</td>
</tr>
<tr>
<td>170</td>
<td>62.5–75.5</td>
</tr>
<tr>
<td>175</td>
<td>66–79</td>
</tr>
<tr>
<td>180</td>
<td>70–83.5</td>
</tr>
<tr>
<td>185</td>
<td>71.5–86.5</td>
</tr>
<tr>
<td>190</td>
<td>78–92.5</td>
</tr>
</tbody>
</table>

Here we want all the dashes in the second column to be vertically aligned, so that we must set them in a separate column; but then there should be no space between the numbers and the dashes connecting them. In such cases we can use the \@ command in the column specification as below

\begin{verbatim}
\begin{center}
\begin{tabular}{|c|c@{--}l|}
\hline
Height (cm) & Ideal weight (kg) \\
\hline
155 & 53.5–64 \\
160 & 56–67 \\
165 & 59–71 \\
170 & 62.5–75.5 \\
175 & 66–79 \\
180 & 70–83.5 \\
185 & 71.5–86.5 \\
190 & 78–92.5 \\
\hline
\end{tabular}
\end{verbatim}
Here the specification \texttt{r@{--}l} indicates that there should be a right aligned column and a left aligned column with a – in between each pair of entries in these columns without the intercolumn space the \texttt{tabular} environment leaves by default between every pair of columns. Note that this incidently saves us the trouble of repeatedly typing \texttt{--}. You can also add some space producing commands within the braces after the \texttt{@} command to produce that much space between the columns on either side of it.

\textbf{VII.2.1. Enhancements to the \texttt{tabular}}

There are many packages which provide further facilities in forming tables. We'll discuss a couple of such packages here.

\textbf{VII.2.2. The \texttt{array} package}

Look at the tables below

\begin{tabular}{|l|c|}
\hline
\multicolumn{1}{|c|}{\centering Planet} & \multicolumn{1}{c|}{{\centering Mean distance from sun \ (km)}} \\
\hline
Mercury & 58100000 \\
Venus & 108300000 \\
Earth & 150000000 \\
Mars & 228250000 \\
Jupiter & 779300000 \\
Saturn & 1429000000 \\
Uranus & 2439000000 \\
Neptune & 4504500000 \\
Pluto & 5903500000 \\
\hline
\end{tabular}

\begin{tabular}{|l|c|}
\hline
\multicolumn{1}{|c|}{\centering Planet} & \multicolumn{1}{c|}{{\centering Mean distance from sun \ (km)}} \\
\hline
Mercury & 58100000 \\
Venus & 108300000 \\
Earth & 150000000 \\
Mars & 228250000 \\
Jupiter & 779300000 \\
Saturn & 1429000000 \\
Uranus & 2439000000 \\
Neptune & 4504500000 \\
Pluto & 5903500000 \\
\hline
\end{tabular}

The one on the right looks nicer, doesn’t it? It was produced using the column specifier \texttt{m} available in the \texttt{array} package. To produce this table, we must first load the \texttt{array} package by the usual \texttt{\usepackage{array}} in the preamble and then type

\begin{verbatim}
\begin{tabular}{|l|c|}
\hline
\multicolumn{1}{|>{\centering\arraybackslash}m{1.5cm}}{\centering Planet} & \multicolumn{1}{m{2.3cm}}{\centering Mean distance from sun \ (km)} \\
\hline
Mercury & 58100000 \\
Venus & 108300000 \\
Earth & 150000000 \\
Mars & 228250000 \\
Jupiter & 779300000 \\
Saturn & 1429000000 \\
Uranus & 2439000000 \\
Neptune & 4504500000 \\
Pluto & 5903500000 \\
\hline
\end{tabular}
\end{verbatim}

The \texttt{m(\textit{wd})} specifier produces a column of width \textit{wd} just like the \texttt{p} specifier, but with the text aligned vertically in the middle unlike the \texttt{p} specifier which aligns the text with the topline. (The table on the left, incidently, was produced by the same input as above but with \texttt{p} instead of \texttt{m}).
Another interesting feature of the \texttt{array} package is the \texttt{\{decl\}} command which can be used before a column specifier. It inserts \texttt{decl} directly in front of the column. For example look at the input below

\begin{verbatim}
\begin{center}
\begin{tabular}{|>{\bfseries}l|r|}
\hline
\multicolumn{1}{|m{1.5cm}|}{\centering Planet} &\multicolumn{1}{m{2.3cm}|}{\centering Mean distance from sun \ \ (km)} \\
\hline
Mercury & 58100000 \\
Venus & 108300000 \\
Earth & 150000000 \\
Mars & 228250000 \\
Jupiter & 779500000 \\
Saturn & 1429000000 \\
Uranus & 2439000000 \\
Neptune & 4504500000 \\
Pluto & 5903500000 \\
\hline
\end{tabular}
\end{center}
\end{verbatim}

which produces the output

\begin{table}
\centering
\begin{tabular}{|>{\bfseries}l|c|}
\hline
Planet & Mean distance from sun (km) \\
\hline
Mercury & 58100000 \\
Venus & 108300000 \\
Earth & 150000000 \\
Mars & 228250000 \\
Jupiter & 779500000 \\
Saturn & 1429000000 \\
Uranus & 2439000000 \\
Neptune & 4504500000 \\
Pluto & 5903500000 \\
\hline
\end{tabular}
\end{table}

The \texttt{array} package also has a \texttt{!} command which works just like the \texttt{@} command, but which does not suppress the intercolumn space.

\subsection{The \texttt{multirow} package}

Look again at the table in 68. Wouldn’t it be nice if the entry “Planet” in the first column is vertically aligned with the center of the two rows in the next column as below?
The package `multirow` is what we need to do this painlessly. It has a command

\texttt{\multirow{\textit{num}}{\textit{wd}}{\textit{item}}}

where \textit{num} is the number of rows to be spanned, \textit{wd} is the width of this column and \textit{item} is the text of the item in this column. This can be used as in the following example

\begin{verbatim}
\begin{center}
\begin{tabular}{|l|r|r|}
\hline
\multirow{3}{1.5cm}{Planet} & \multicolumn{2}{p{3.5cm}|}{\centering Distance from sun \ (million km)} \\
& \multicolumn{1}{c|}{Maximum} & \multicolumn{1}{c|}{Minimum} \\
\hline
Mercury & 69.4 & 46.8 \\
Venus & 109.0 & 107.6 \\
Earth & 152.6 & 147.4 \\
Mars & 249.2 & 207.3 \\
Jupiter & 817.4 & 741.6 \\
Saturn & 1512.0 & 1346.0 \\
Uranus & 3011.0 & 2740.0 \\
Neptune & 4543.0 & 4466.0 \\
Pluto & 7346.0 & 4461.0 \\
\hline
\end{tabular}
\end{center}
\end{verbatim}

But this code does not produce the table above, but only
VII.2. Tables

<table>
<thead>
<tr>
<th>Planet</th>
<th>Distance from sun (million km)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maximum</td>
<td>Minimum</td>
</tr>
<tr>
<td>Mercury</td>
<td>69.4</td>
<td>46.8</td>
</tr>
<tr>
<td>Venus</td>
<td>109.0</td>
<td>107.6</td>
</tr>
<tr>
<td>Earth</td>
<td>152.6</td>
<td>147.4</td>
</tr>
<tr>
<td>Mars</td>
<td>249.2</td>
<td>207.3</td>
</tr>
<tr>
<td>Jupiter</td>
<td>817.4</td>
<td>741.6</td>
</tr>
<tr>
<td>Saturn</td>
<td>1512.0</td>
<td>1346.0</td>
</tr>
<tr>
<td>Uranus</td>
<td>3011.0</td>
<td>2740.0</td>
</tr>
<tr>
<td>Neptune</td>
<td>4543.0</td>
<td>4466.0</td>
</tr>
<tr>
<td>Pluto</td>
<td>7346.0</td>
<td>4461.0</td>
</tr>
</tbody>
</table>

The trouble is that though the entry “Planet” is vertically centered in its column, it is not horizontally centered. The horizontal alignment is controlled by the command \multirowsetup and this is by default set to \raggedright. So all that is needed to get the beautiful table seen at the beginning of this section is to add the line
\renewcommand{\multirowsetup}{\centering}
at the beginning of the code above.

VII.2.4 tabbing vs. tabular

Let’s take a quick look at the pros and cons of the tabbing and tabular environments.

- The tabbing environment can be typeset only as a separate paragraph, while the tabular environment can be placed anywhere in text, even inside Mathematics.
- The tabbing environment can span multiple pages, but the tabular environment cannot.
- tabbing environments cannot be nested, while tabular environments can be nested to any number of levels.

VII.2.5 Multipage tables—The package longtable

As we have noted, we cannot create table spanning more than one page using the tabular environment. But the package longtable by David Carlisle can do this and it has quite a few other tricks also. To use this package, load it as usual with the command \usepackage{longtable} in the preamble and then to produce a no-frills “longtable” just use the commands \begin{longtable} ... \end{longtable} instead of the \begin{tabular} ... \end{tabular} commands. We can use footnotes and the \newpage commands inside the longtable environment. If the package array is also loaded, its extra features can be used.

Apart from this, this package has provisions to specify at the start of the input the following items

- the rows that should appear at the top of the table; the input for these to be terminated by \endfirsthead
- the rows that should appear in every page after the first; such input terminated by \endhead
- those at the bottom of every page; the input terminated by \endfoot
those rows at the *very end* of the table, terminated by `\endlastfoot` These are illustrated in the (long!) table below.

### Science and Technology in the Twentieth Century

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1900</td>
<td>Max Planck proposes quantum theory</td>
</tr>
<tr>
<td></td>
<td>Publication of Sigmund Freud’s <em>The Interpretation of Dreams</em></td>
</tr>
<tr>
<td>1901</td>
<td>Discovery of principal blood groups</td>
</tr>
<tr>
<td></td>
<td>Guglielmo Marconi transmits wireless signals across the Atlantic</td>
</tr>
<tr>
<td>1903</td>
<td>Wright brothers make their first flight</td>
</tr>
<tr>
<td>1905</td>
<td>Albert Einstein presents Special Theory of Relativity</td>
</tr>
<tr>
<td>1911</td>
<td>Ernest Rutherford proposes theory of atomic structure</td>
</tr>
<tr>
<td>1912</td>
<td>Victor Hess discovers cosmic rays</td>
</tr>
<tr>
<td>1916</td>
<td>Albert Einstein presents general Theory of Relativity</td>
</tr>
<tr>
<td>1920</td>
<td>Radio broadcasting begins</td>
</tr>
<tr>
<td>1926</td>
<td>John Logie Baird demonstrates television</td>
</tr>
<tr>
<td>1928</td>
<td>Alexander Fleming discovers penicillin</td>
</tr>
<tr>
<td>1933</td>
<td>Discovery of polythene</td>
</tr>
<tr>
<td>1934</td>
<td>Discovery of nuclear fission</td>
</tr>
<tr>
<td>1938</td>
<td>Discovery of nylon</td>
</tr>
<tr>
<td>1940</td>
<td>Plutonium obtained by bombardment of uranium</td>
</tr>
<tr>
<td>1942</td>
<td>Construction of first nuclear reactor</td>
</tr>
<tr>
<td>1946</td>
<td>Construction of first electronic digital computer</td>
</tr>
<tr>
<td>1947</td>
<td>First supersonic flight</td>
</tr>
<tr>
<td></td>
<td>Invention of the transistor</td>
</tr>
<tr>
<td>1951</td>
<td>Nuclear power stations introduced</td>
</tr>
<tr>
<td>1953</td>
<td>James Watson and Francis Crick show DNA molecule structure</td>
</tr>
<tr>
<td>1956</td>
<td>Contraceptive pill introduced</td>
</tr>
<tr>
<td>1957</td>
<td>Launch of the first space satellite (<em>Sputnik 1</em>)</td>
</tr>
<tr>
<td>1959</td>
<td>First photograph of the dark side of the moon (<em>Luna 3</em>)</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>...</td>
<td>............</td>
</tr>
<tr>
<td>1961</td>
<td>Yuri Gagarin becomes first man in space (<em>Vostok 1</em>)</td>
</tr>
<tr>
<td>1966</td>
<td>First lunar soft landing (<em>Luna 9</em>)</td>
</tr>
<tr>
<td>1967</td>
<td>Discovery of pulsars</td>
</tr>
<tr>
<td>1968</td>
<td>First manned lunar orbit (<em>Apollo 8</em>)</td>
</tr>
<tr>
<td>1969</td>
<td>First man on moon (Neil Armstrong)</td>
</tr>
<tr>
<td>1972</td>
<td>Pocket calculator introduced</td>
</tr>
</tbody>
</table>

*continued on the next page*
Science and Technology in the Twentieth Century (continued)

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1974</td>
<td>First ‘test-tube babies’</td>
</tr>
<tr>
<td>1977</td>
<td>Launch of <em>Voyager</em> missions to outer space</td>
</tr>
<tr>
<td>1983</td>
<td>IBM personal computer launched</td>
</tr>
<tr>
<td>1986</td>
<td>Hailey’s comet intercepted</td>
</tr>
<tr>
<td>1997</td>
<td>Cloning of “Dolly” the sheep</td>
</tr>
<tr>
<td>2000</td>
<td>Decoding of 90% of human genome completed</td>
</tr>
</tbody>
</table>

Source: *The Cambridge Factfinder*

Part of the code to produce this is given below.

```
\renewcommand{\arraystretch}{1.2}
\begin{longtable}{|c|l|}
\multicolumn{2}{c}%%
\hline
\textbf{Science and Technology in the Twentieth Century}\\[5pt]
\hline
\textbf{Year} & \textbf{Event} \\
\hline
1974 & First ‘test-tube babies’ \\
1977 & Launch of *Voyager* missions to outer space \\
1983 & IBM personal computer launched \\
1986 & Hailey’s comet intercepted \\
1997 & Cloning of “Dolly” the sheep \\
2000 & Decoding of 90% of human genome completed \\
\hline
\endfoot
\hline
\textit{continued on the next page}\\[5pt]
\endlastfoot
\hline
\textbf{Science and Technology in the Twentieth Century}\\[5pt]
\textit{(continued)}
\endfirsthead
\hline
\textbf{Year} & \textbf{Event} \\
\hline
1900 & Max Planck proposes quantum theory \\
1990 & Decoding of 90% of human genome completed \\
\hline
\hline
\endhead
\hline
\textit{continued on the next page}\\[5pt]
\endfoot
\hline
\textbf{Science and Technology in the Twentieth Century}\\[5pt]
\textit{continued on the next page}\\[5pt]
\endlastfoot
\hline
\end{longtable}
```

VII.2.6. And that’s not all!

There are many more packages which help to produce tables of various requirements. Be sure to check out the packages `tabularx`, `delarray`, `dcolumn` and `hhline`. 
Donal Knuth created TeX primarily to typeset Mathematics beautifully. \TeX\ includes all the capabilities of \TeX\ in Mathematics typesetting, sometimes with easier user interfaces. Then there are packages like amsmath which enhance and refine these interfaces.

\section{The Basics}

A mathematical expression occurring in running text (called \textit{in-text} math) is produced by enclosing it between dollar signs. Thus to produce

\begin{quote}
The equation representing a straight line in the Cartesian plane is of the form $ax+by+c=0$, where $a$, $b$, $c$ are constants.
\end{quote}

we type

\begin{quote}
The equation representing a straight line in the Cartesian plane is of the form $ax+by+c=0$, where $a$, $b$, $c$ are constants.
\end{quote}

Some comments are in order. First note that the text within dollars is typeset in \textit{italic} (actually math italic). Again, even though we did not leave any spaces within $ax+by+c=0$, \TeX\ leaves spaces on either side of the addition signs and the equality sign. On the other hand, even if we type $ax + by + c = 0$, the output would be the same: $ax + by + c = 0$. The moral: \TeX\ has its own spacing rules in math mode.

To see another instance of this, change the last part of the code above to read

\begin{quote}
... where $a$, $b$, $c$ are constants.
\end{quote}

Saves some typing, does not it? But look at the output.

\begin{quote}
The equation representing a straight line in the Cartesian plane is of the form $ax+by+c=0$, where $a$, $b$, $c$ are constants.
\end{quote}

Do you see the difference? There are no spaces after the commas, though we had such spaces in the output. So \TeX\ swallows spaces in math mode (you can not save dollars that way!).

Incidentally, dollar signs are \TeX\ way of distinguishing Mathematical text. \TeX\ has other ways also of doing it, using \textbackslash( ... \textbackslash) or \textbackslash\begin{math} ... \end{math}. Thus either of the inputs shown below also produces the same output as above.

\begin{quote}
The equation representing a straight line in the Cartesian plane is of the form \textbackslash(ax+by+c=0\textbackslash), where \textbackslash(a\textbackslash), \textbackslash(b\textbackslash), \textbackslash(c\textbackslash) are constants.
\end{quote}

\begin{quote}
The equation representing a straight line in the Cartesian plane is of the form \textbackslash\begin{math}ax+by+c=0\end{math}, where \textbackslash\begin{math}a\end{math} a \textbackslash\end{math}, \textbackslash\begin{math}b\end{math} b \textbackslash\end{math}, \textbackslash\begin{math}c\end{math} c \textbackslash\end{math} are constants.
\end{quote}
Now suppose we want to display the equation in the above output as in

The equation representing a straight line in the Cartesian plane is of the form

\[ ax + by + c = 0 \]

where \( a, b, c \) are constants.

This can be done by changing the input as follows:

The equation representing a straight line in the Cartesian plane is of the form

\[
ax + by + c = 0
\]

where \( a, b, c \) are constants.

Again \( \ldots \) \( \ldots \) is the \( \LaTeX \) way of producing displayed math. \( \LaTeX \) has the constructs \( \left[ \ldots \right] \) or \( \begin{displaymath} \ldots \end{displaymath} \) also to do this.

VIII.1.1. Superscripts and subscripts

Look at the text below

In the seventeenth century, Fermat conjectured that if \( n > 2 \), then there are no integers \( x, y, z \) for which

\[ x^n + y^n = z^n. \]

This was proved in 1994 by Andrew Wiles.

This is produced by the input

In the seventeenth century, Fermat conjectured that if \( n > 2 \), then there are no integers \( x, y, z \) for which

\[
 x^n + y^n = z^n.
\]

This was proved in 1994 by Andrew Wiles.

This shows that superscripts (mathematicians call them exponents) are produced by the \(^ \) symbol. If the superscript is more than one character long, we must be careful to group these characters properly. Thus to produce

\[ (x^m)^n = x^{mn}. \]

we must type

\[ (x^m)^n = x^{mn}. \]

Instead of \( x^{(mn)} \), if we type \( x^{mn} \) we end up with \( x^{mn} \) instead of the intended \( x^{(mn)} \) in the output.

We can have superscripts of superscripts (and mathematicians do need them). For example,

Numbers of the form \( 2^{2^n} + 1 \), where \( n \) is a natural number, are called Fermat numbers.

is produced by
Numbers of the form $2^{2^n}+1$, where $n$ is a natural number, are called Fermat numbers.

Note the grouping of superscripts. (What happens if you type $2^{2^n}+1$ or $(2^2)^n$?)

Now let us see how subscripts (mathematicians call them subscripts) are produced.

The sequence \((x_n)\) defined by

\[
x_1 = 1, \quad x_2 = 1, \quad x_n = x_{n-1} + x_{n-2} \quad (n > 2)
\]

is called the Fibonacci sequence.

Thus subscripts are produced by the \_ character. Note how we insert spaces by the \quad command. (The command \; in \textit{math mode} produces what is known as a “thickspace”.)

Subscripts of subscripts can be produced as in the case of superscripts (with appropriate grouping).

We can also have superscripts and subscripts together. Thus

\[
\text{If the sequence } (x_n) \text{ converges to } a, \text{ then the sequence } (x_n^2) \text{ converges to } a^2
\]

is produced by

\[
\text{If the sequence } (x_n) \text{ converges to } a, \text{ then the sequence } (x_n^2) \text{ converges to } a^2
\]

Again, we must be careful about the grouping (or the lack of it) when typesetting superscripts and subscripts together. The following inputs and the corresponding outputs make the point.

\[
x_{m}^n \quad x_{m}^n \quad x_{m}^n \quad x_{m}^n
\]

(This has to do with the way \textsc{TeX} works, producing “boxes” to fit the output characters. The box for \(x_m^n\) is like \includegraphics[width=0.5cm]{box.png} while the box for \(x_m^m\) is \includegraphics[width=0.5cm]{box.png}.)

\textbf{VIII.1.2. Roots}

Square roots are produced by the \texttt{\sqrt} argument. Thus $\sqrt{2}$ produces $\sqrt{2}$. This command has an optional argument to produce other roots. Thus

\[
\text{Which is greater } \sqrt{5} \text{ or } \sqrt{4}?
\]

is produced by
Which is greater $\sqrt[4]{5}$ or $\sqrt[5]{4}$?

The horizontal line above the root (called vinculum by mathematicians of yore) elongates to accommodate the enclosed text. For example, $\sqrt{x+y}$ produces $\sqrt{x+y}$.

Also, you can produce nested roots as in

$$2 \sqrt{2}, \quad 2^2 \sqrt{2 - \sqrt{2}}, \quad 2^3 \sqrt{2 - \sqrt{2 + \sqrt{2}}}, \quad 2^4 \sqrt{2 - \sqrt{2 + \sqrt{2 + \sqrt{2}}}}, \ldots$$

converge to $\pi$.

by typing

The sequence

\[
2 \sqrt{2}, \quad 2^2 \sqrt{2 - \sqrt{2}}, \quad 2^3 \sqrt{2 - \sqrt{2 + \sqrt{2}}}, \quad 2^4 \sqrt{2 - \sqrt{2 + \sqrt{2 + \sqrt{2}}}}, \ldots
\]

The $\ldots$ command above produces ..., the three dots indicating indefinite continuation, called ellipsis (more about them later). The command $\backslash$, produces a “thinspace” (as opposed to a thickspace produced by $\backslash ;$, seen earlier). Why all this thin and thick spaces in the above input? Remove them and see the difference. (A tastefully applied thinspace is what makes a mathematical expression typeset in $\LaTeX$ really beautiful.)

The symbol $\pi$ in the output produced by $\pi$ maybe familiar from high school mathematics. It is a Greek letter named “pi”. Mathematicians often use letters of the Greek alphabet (which even otherwise is Greek to many) and a multitude of other symbols in their work. A list of available symbols in $\LaTeX$ is given at the end of this chapter.

VIII.1.3. Mathematical symbols

In the list at the end of this chapter, note that certain symbols are marked to be not available in native $\LaTeX$, but only in certain packages. We will discuss some such packages later. Another thing about the list is that they are categorized into classes such as “Binary Relations”, “Operators”, “Functions” and so on. This is not merely a matter of convenience.

We have noted that $\TeX$ leaves some additional spaces around “binary operators” such as $+$ and $-$. The same is true for any symbol classified as a binary operator. For example, consider the following

For real numbers $x$ and $y$, define an operation $\circ$ by

$$x \circ y = x + y - xy$$

This operation is associative.

From the list of symbols, we see that $\circ$ is produced by $\circ$ and this is classified as a binary operator, so that we can produce this by

For real numbers $x$ and $y$, define an operation $\circ$ by

$$x \circ y = x + y - xy$$
\(x \circ y = x+y-xy\)

\[x \circ y = x+y-xy\]

This operation is associative.

Note the spaces surrounding the \(\circ\) symbol in the output. On the other hand suppose you want

For real numbers \(x\) and \(y\), define an operation \(\Box\) by

\[x \Box y = x^2 + y^2\]

The list of symbols show that the symbol \(\Box\) is produced by \texttt{\Box} but that it is available only in the package \texttt{latexsym} or \texttt{amssymb}. So if we load one of these using the \texttt{\usepackage} command and then type

For real numbers \(x\) and \(y\), define an operation \(\Box\) by

\[x \Box y = x^2 + y^2\]

you will only get

For real numbers \(x\) and \(y\), define an operation \(\Box\) by

\[x \Box y = x^2 + y^2\]

Notice the difference? There are no spaces around \(\Box\); this is because, this symbol is not by default defined as a binary operator. (Note that it is classified under “Miscellaneous”.) But we can ask \TeX{} to consider this symbol as a binary operator by the command \texttt{\mathbin} before \texttt{\Box} as in

For real numbers \(x\) and \(y\), define an operation \(\Box\) by

\[x \mathbin \Box y = x^2 + y^2\]

and this will produce the output shown first.

This holds for “Relations” also. \TeX{} leaves some space around “Relation” symbols and we can instruct \TeX{} to consider any symbol as a relation by the command \texttt{\mathrel}.

Thus we can produce

Define the relation \(\rho\) on the set of real numbers by \(x \mathrel \rho y\) iff \(x-y\) is a rational number.

by typing

Define the relation \(\rho\) on the set of real numbers by \(x \mathrel \rho y\) iff \(x-y\) is a rational number.

(See what happens if you remove the \texttt{\mathrel} command.)

\section{Custom commands}

We have seen that \LaTeX{} produces mathematics (and many other things as well) by means of “commands”. The interesting thing is that we can build our own commands using the ones available. For example, suppose that the expression \((x_1, x_2, \ldots, x_n)\) occurs frequently in a document. If we now write
\newcommand{\vect}{(x_1,x_2,\ldots,x_n)}

Then we can type $\vect$s anywhere afterwards to produce $(x_1,x_2,\ldots,x_n)$ as in

We often write $x$s to denote the vector $\vect$.

to get

We often write $x$ to denote the vector $(x_1,x_2,\ldots,x_n)$.

(By the way, the best place to keep such “newcommands” is the preamble, so that you can use them anywhere in the document. Also, it will be easier to change the commands, if the need arises).

OK, we can now produce $(x_1,x_2,\ldots,x_n)$ with $\vect$s, but how about $(y_1,y_2,\ldots,y_n)$ or $(z_1,z_2,\ldots,z_n)$? Do we have to define newcommands for each of these? Not at all. We can also define commands with \textit{variable arguments} also. Thus if we change our definition of $\vect$ to

\newcommand{\vect}[1]{(#1_1,#1_2,\dotsc,#1_n)}

Then we can use $\vect(x)$ to produce $(x_1,x_2,\ldots,x_n)$ and $\vect(a)$ to produce $(a_1,a_2,\ldots,a_n)$ and so on.

The form of this definition calls for some comments. The [1] in the \newcommand above indicates that the command is to have \textit{one} (variable) argument. What about the #1? Before producing the output, each occurrence of #1 will be replaced by the (single) argument we supply to \vect in the input. For example, the input $\vect(a)$ will be changed to $(a_1,a_2,\dotsc,a_n)$ at some stage of the compilation.

We can also define commands with more than one argument (the maximum number is 9). Thus for example, if the document contains not only $(x_1,x_2,\ldots,x_n)$, $(y_1,y_2,\ldots,y_n)$ and so on, but $(x_1,x_2,\ldots,x_m)$, $(y_1,y_2,\ldots,y_p)$ also, then we can change our definition of $\vect$ to

\newcommand{\vect}[2]{(#1_1,#1_2,\dotsc,#1_#2)}

so that we can use $\vect(x,n)$ to produce $(x_1,x_2,\ldots,x_n)$ and $\vect(a,p)$ to produce $(a_1,a_2,\ldots,a_p)$.

\section*{VIII.3. More on Mathematics}
There are some many other features of typesetting math in \LaTeX, but these have better implementations in the package amsmath which has some additional features as well. So, for the rest of the chapter the discussion will be with reference to this package and some allied ones. Thus all discussion below is under the assumption that the package amsmath has been loaded with the command $\textbackslash$usepackage{amsmath}.

\subsection*{VIII.3.1. Single equations}
In addition to the \LaTeX commands for displaying math as discussed earlier, the amsmath also provides the \texttt{\begin{equation*} \ldots \end{equation*}} construct. Thus with this package loaded, the output

\begin{equation*}
  ax + by + c = 0
\end{equation*}

where $a$, $b$, $c$ are constants.

can also be produced by
The equation representing a straight line in the Cartesian plane is of the form
\begin{equation*} ax + by + c = 0 \end{equation*}
where $a$, $b$, $c$ are constants.

Why the \* after equation? Suppose we try it without the \* as
\begin{equation}
ax + by + c = 0
\end{equation}
where $a$, $b$, $c$ are constants.

we get

\begin{equation*}
\text{(VIII.1) } ax + by + c = 0
\end{equation*}

where $a$, $b$, $c$ are constants.

This provides the equation with a number. We will discuss equation numbering in some more detail later on. For the time being, we just note that for any environment name with a star we discuss here, the unstarred version provides the output with numbers.

Ordinary text can be inserted inside an equation using the \text command. Thus we can get

\begin{equation*}
\text{Thus for all real numbers } x \text{ we have }

x \leq |x| \quad \text{and} \quad x \geq |x|
\end{equation*}

and so

\begin{equation*}
x \leq |x| \quad \text{for all } x \text{ in } R.
\end{equation*}

from

\begin{equation*}
\text{Thus for all real numbers } x \text{ we have }
\begin{align*}
x \leq |x| & \quad \text{and} \quad |x| \leq |x|
\end{align*}
\end{equation*}

\begin{equation*}
\text{and so }
\begin{align*}
x \leq |x| & \quad \text{for all } x \text{ in } R.
\end{align*}
\end{equation*}

Note the use of dollar signs in the second \text above to produce mathematical symbols within \text.

Sometimes a single equation maybe too long to fit into one line (or sometimes even two lines). Look at the one below:

\begin{equation*}
(a + b + c + d + e)^2 = a^2 + b^2 + c^2 + d^2 + e^2
+ 2ab + 2ac + 2ad + 2ae + 2bc + 2bd + 2be + 2cd + 2ce + 2de
\end{equation*}
This is produced by the environment `multline*` (note the spelling carefully—it is *not* `mult\line*`), as shown below.

\begin{multline*}
(a+b+c+d+e)^2 = a^2 + b^2 + c^2 + d^2 + e^2 + 2ab + 2ac + 2ad + 2ae + 2bc + 2bd + 2be + 2cd + 2ce + 2de
\end{multline*}

`multline` can be used for equations requiring more than two lines, but without tweaking, the results are not very satisfactory. For example, the input

\begin{multline*}
(a+b+c+d+e+f)^2 = a^2 + b^2 + c^2 + d^2 + e^2 + f^2 + 2ab + 2ac + 2ad + 2ae + 2af + 2bc + 2bd + 2be + 2bf + 2cd + 2ce + 2cf + 2de + 2df + 2ef
\end{multline*}

produces

\[
(a + b + c + d + e + f)^2 = a^2 + b^2 + c^2 + d^2 + e^2 + f^2 + 2ab + 2ac + 2ad + 2ae + 2af + 2bc + 2bd + 2be + 2bf + 2cd + 2ce + 2cf + 2de + 2df + 2ef
\]

By default, the `multline` environment places the first line flush left, the last line flush right (except for some indentation) and the lines in between, centered within the display.

A better way to typeset the above `multline` (not `multline`) equation is as follows.

\begin{equation*}
\begin{split}
(a+b+c+d+e+f)^2 &= a^2 + b^2 + c^2 + d^2 + e^2 + f^2 + 2ab + 2ac + 2ad + 2ae + 2af + 2bc + 2bd + 2be + 2bf + 2cd + 2ce + 2cf + 2de + 2df + 2ef
\end{split}
\end{equation*}

This is done using the `split` environment as shown below.

\begin{equation*}
\begin{split}
(a+b+c+d+e+f)^2 &= a^2 + b^2 + c^2 + d^2 + e^2 + f^2 + 2ab + 2ac + 2ad + 2ae + 2af + 2bc + 2bd + 2be + 2bf + 2cd + 2ce + 2cf + 2de + 2df + 2ef
\end{split}
\end{equation*}
Some comments seems to be in order. First note that the `split` environment cannot be used independently, but only inside some equation structure such as `equation` (and others we will soon see). Unlike `multline`, the `split` environment provides for alignment among the “split” lines (using the `&` character, as in `tabular`). Thus in the above example, all the `+` signs are aligned and these in turn are aligned with a point `\quad` to the right of the `=` sign. It is also useful when the equation contains multiple equalities as in

\[
(a + b)^2 = (a + b)(a + b) \\
= a^2 + ab + ba + b^2 \\
= a^2 + 2ab + b^2
\]

which is produced by

\begin{equation*}
\begin{split}
(a+b)^2 & = (a+b)(a+b) \\
& = a^2+ab+ba+b^2 \\
& = a^2+2ab+b^2 \\
\end{split}
\end{equation*}

\textbf{VIII.3.2. Groups of equations}

A group of displayed equations can be typeset in a single go using the `gather` environment. For example,

\[(a,b)+(c,d)=(a+c,b+d)\]
\[(a,b)(c,d)=(ac-bd,ad+bc)\]

can be produced by

\begin{gather*}
(a,b)+(c,d)=(a+c,b+d) \\
(a,b)(c,d)=(ac-bd,ad+bc)
\end{gather*}

Now when several equations are to be considered one unit, the logically correct way of typesetting them is with some alignment (and it is perhaps easier on the eye too). For example,

Thus \(x\), \(y\) and \(z\) satisfy the equations

\[
x + y - z = 1 \\
x - y + z = 1
\]

This is obtained by using the `align*` environment as shown below

\begin{align*}
\text{Thus } & x, y \text{ and } z \text{ satisfy the equations} \\
& x + y - z = 1 \\
& x - y + z = 1
\end{align*}
We can add a short piece of text between the equations, without disturbing the alignment, using the \texttt{\inter} command. For example, the output

<table>
<thead>
<tr>
<th>Thus $x$, $y$ and $z$ satisfy the equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x + y - z = 1$</td>
</tr>
<tr>
<td>$x - y + z = 1$</td>
</tr>
<tr>
<td>and by hypothesis</td>
</tr>
<tr>
<td>$x + y + z = 1$</td>
</tr>
</tbody>
</table>

is produced by

\begin{align*}
\text{Thus } s \text{, } t \text{ and } u \text{ satisfy the equations} \\
\begin{align}
x+y-z & = 1 \\
x-y+z & = 1 \\
\text{and by hypothesis} \\
x+y+z & = 1 \\
\end{align}
\end{align*}

We can also set multiple ‘columns’ of aligned equations side by side as in

<table>
<thead>
<tr>
<th>Compare the following sets of equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\cos^2 x + \sin^2 x = 1$</td>
</tr>
<tr>
<td>$\cos^2 x - \sin^2 x = \cos 2x$</td>
</tr>
<tr>
<td>$\cosh^2 x - \sinh^2 x = 1$</td>
</tr>
<tr>
<td>$\cosh^2 x + \sinh^2 x = \cosh 2x$</td>
</tr>
</tbody>
</table>

All that it needs are extra &’s to separate the columns as can be seen from the input

\begin{align*}
\text{Compare the following sets of equations} \\
\begin{align}
\cos^2 x + \sin^2 x & = 1 & \cosh^2 x - \sinh^2 x & = 1 \\
\cos^2 x - \sin^2 x & = \cos 2x & \cosh^2 x + \sinh^2 x & = \cosh 2x \\
\end{align}
\end{align*}

We can also adjust the horizontal space between the equation columns. For example,

\begin{align*}
\text{Compare the sets of equations} \\
\begin{align}
\cos^2 x + \sin^2 x & = 1 & \cosh^2 x - \sinh^2 x & = 1 \\
\cos^2 x - \sin^2 x & = \cos 2x & \cosh^2 x + \sinh^2 x & = \cosh 2x \\
\end{align}
\end{align*}

gives

<table>
<thead>
<tr>
<th>Compare the sets of equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\cos^2 x + \sin^2 x = 1$</td>
</tr>
<tr>
<td>$\cos^2 x - \sin^2 x = \cos 2x$</td>
</tr>
<tr>
<td>$\cosh^2 x - \sinh^2 x = 1$</td>
</tr>
<tr>
<td>$\cosh^2 x + \sinh^2 x = \cosh 2x$</td>
</tr>
</tbody>
</table>

Perhaps a nicer way of typesetting the above is
Compare the following sets of equations

\[
\begin{align*}
\cos^2 x + \sin^2 x &= 1 \\
\cos^2 x - \sin^2 x &= \cos 2x
\end{align*}
\quad \text{and} \quad
\begin{align*}
\cosh^2 x - \sinh^2 x &= 1 \\
\cosh^2 x + \sinh^2 x &= \cosh 2x
\end{align*}
\]

This cannot be produced by the equation structures discussed so far, because any of these environments takes up the entire width of the text for its display, so that we cannot put anything else on the same line. So amsmath provides variants \texttt{gathered}, \texttt{aligned} and \texttt{alignedat} which take up only the \textit{actual width of the contents} for their display. Thus the above example is produced by the input

\begin{equation*}
\begin{aligned}
\cos^2 x + \sin^2 x & = 1 \\
\cos^2 x - \sin^2 x & = \cos 2x
\end{aligned}
\qquad \text{and} \quad
\begin{aligned}
\cosh^2 x - \sinh^2 x & = 1 \\
\cosh^2 x + \sinh^2 x & = \cosh 2x
\end{aligned}
\end{equation*}

Another often recurring structure in mathematics is a display like this

\[
|x| = \begin{cases} 
  x & \text{if } x \geq 0 \\
  -x & \text{if } x \leq 0
\end{cases}
\]

There is a special environment \texttt{cases} in amsmath to take care of these. The above example is in fact produced by

\begin{equation*}
|x| = \begin{cases}
  x & \text{if } x \geq 0 \\
  -x & \text{if } x \leq 0
\end{cases}
\end{equation*}

\section{Numbered equations}

We have mentioned that each of the the ‘starred’ equation environments has a corresponding unstarred version, which also produces numbers for their displays. Thus our very first example of displayed equations with \texttt{equation} instead of \texttt{equation*} as in

\begin{equation}
ax + by + c = 0
\end{equation}

where $a$, $b$, $c$ are constants.
The equation representing a straight line in the Cartesian plane is of the form

\begin{equation}
ax + by + c = 0
\end{equation}

where \(a, b, c\) are constants.

Why \(\text{VIII.2}\) for the equation number? Well, this is Equation number 2 of Chapter \(\text{VIII}\), isn’t it? If you want the section number also in the equation number, just give the command

\begin{verbatim}
\numberwithin{equation}{section}
\end{verbatim}

We can also override the number \(L\) \LaTeX\ produces with one of our own design with the \texttt{\tag}\ command as in

\begin{equation}
ax + by + c = 0 \tag{L}
\end{equation}

which gives

The equation representing a straight line in the Cartesian plane is of the form

\begin{equation}
ax + by + c = 0
\end{equation}

where \(a, b, c\) are constants.

There is also a \texttt{\tag*}\ command which typesets the \textit{equation label} without parentheses.

What about numbering alignment structures? Except for \texttt{split} and \texttt{aligned}, all other alignment structures have unstarred forms which attach numbers to \textit{each} aligned equation. For example,

\begin{verbatim}
\begin{align}
x+y-z & = 1 \\
x-y+z & = 1
\end{align}
\end{verbatim}

gives

\begin{align}
(VIII.3) & \quad x + y - z = 1 \\
(VIII.4) & \quad x - y + z = 1
\end{align}

Here is also, you can give a label of your own to \textit{any} of the equations with the \texttt{\tag}\ command. Be careful to give the \texttt{\tag} \textit{before} the end of line character \texttt{\\\} though. (See what happens if you give a \texttt{\tag} \textit{after} a \texttt{\\\}.) You can also suppress the label for any equation with the \texttt{\notag}\ command. These are illustrated in the sample input below:

Thus \(x, y\) and \(z\) satisfy the equations

\begin{verbatim}
\begin{align*}
\end{verbatim}

\[ \begin{align*}
  x+y-z &= 1 
  \tag{} \\
  x-y+z &= 1 
  \notag \\
  \intertext{and by hypothesis} 
  x+y+z &= 1 \tag{H} 
  \end{align*} \]

which gives the following output

Thus \( x, y \) and \( z \) satisfy the equations

\[
\begin{align*}
  x + y - z &= 1 \\
  x - y + z &= 1 
\end{align*}
\]

and by hypothesis

\( \begin{equation}
  x + y + z = 1
\end{equation} \)

What about \texttt{split} and \texttt{aligned}? As we have seen, these can be used only within some other equation structure. The numbering or the lack of it is determined by this parent structure. Thus

\[
\begin{align*}
  (a+b)^2 &= (a+b)(a+b) \\
  &= a^2 + ab + ba + b^2 \\
  &= a^2 + 2ab + b^2
\end{align*}
\]

There are more things Mathematics than just equations. Let us look at how \LaTeX{} and in particular, the \texttt{amsmath} package deals with them.

\section*{VIII.4. Matrices}

Matrices are by definition numbers or mathematical expressions arranged in rows and columns. The \texttt{amsmath} has several environments for producing such arrays. For example
The system of equations

\begin{align*}
x + y - z &= 1 \\
x - y + z &= 1 \\
x + y + z &= 1
\end{align*}

can be written in matrix terms as

\[
\begin{pmatrix}
1 & 1 & -1 \\
1 & -1 & 1 \\
1 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix}
= \begin{pmatrix}
1 \\
1 \\
1
\end{pmatrix}.
\]

Here, the matrix \[
\begin{pmatrix}
1 & 1 & -1 \\
1 & -1 & 1 \\
1 & 1 & 1
\end{pmatrix}
\] is invertible.

Note that the environment \texttt{pmatrix} can be used within in-text mathematics or in displayed math. Why the \texttt{p}? There is indeed an environment \texttt{matrix} (without a \texttt{p}) but it
produces an array without the enclosing parentheses (try it). If you want the array to be enclosed within square brackets, use \texttt{bmatrix} instead of \texttt{pmatrix}. Thus

\begin{pmatrix}
a & b \\ 
c & d \\
\end{pmatrix}

is produced by

\begin{verbatim}
\begin{pmatrix}
a & b \\ 
c & d \\
\end{pmatrix}
\end{verbatim}

while others prefer square brackets as in

\begin{bmatrix}
a & b \\ 
c & d \\
\end{bmatrix}

There is also a \texttt{vmatrix} environment, which is usually used for determinants as in

\begin{vmatrix}
\begin{vmatrix}
a & b \\ 
c & d \\
\end{vmatrix}
\end{vmatrix}

which is obtained from the input

\begin{verbatim}
\begin{vmatrix}
a & b \\ 
c & d \\
\end{vmatrix}
\end{verbatim}

There is a variant \texttt{Vmatrix} which encloses the array in double lines. Finally, we have a \texttt{Bmatrix} environment which produces an array enclosed within braces \{\}.
A row of dots in a matrix can be produced by the command \texttt{\textbackslash hdotsfour}. It should be used with an argument specifying the number of columns to be spanned. For example, to get

A general $m \times n$ matrix is of the form

\[
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\hdotsfor[4]{} \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{pmatrix}
\]

we type

A general $m \times n$ matrix is of the form
\begin{equation}
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\hdotsfor{4} \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{pmatrix}
\end{equation}

The command \texttt{\textbackslash hdotsfor} has also an optional argument to specify the spacing of dots. Thus in the above example, if we use \texttt{\textbackslash hdotsfor[2]{4}}, then the space between the dots is doubled as in

A general $m \times n$ matrix is of the form
\begin{equation}
\begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\hdotsfor{2}{4} \\
a_{m1} & a_{m2} & \cdots & a_{mn}
\end{pmatrix}
\end{equation}

\textbf{VIII.4.2. Dots}

In the above example, we used the command \texttt{\textbackslash dots} to produce a row of three dots. This can be used in other contexts also. For example,

Consider a finite sequence $X_1, X_2, \dotsc$, its sum $X_1 + X_2 + \dotsc$ and product $X_1 X_2 \dotsc$.

gives

Consider a finite sequence $X_1, X_2, \ldots$, its sum $X_1 + X_2 + \ldots$ and product $X_1 X_2 \ldots$.

Here the dots in all the three contexts are along the “baseline” of the text. Isn’t it better to typeset this as

Consider a finite sequence $X_1, X_2, \ldots$, its sum $X_1 + X_2 + \cdots$ and product $X_1 X_2 \cdots$.

with \textit{raised} dots for addition and multiplication? The above text is typeset by the input

Consider a finite sequence $X_1, X_2, \texttt{\textbackslash dotsc}$, its sum $X_1 + X_2 + \texttt{\textbackslash dotsb}$ and product $X_1 X_2 \texttt{\textbackslash dotsm}$.
Here \( \texttt{\textbackslash dotsc} \) stands for dots to be used with commas, \( \texttt{\textbackslash dotsb} \) for dots with binary operations (or relations) and \( \texttt{\textbackslash dotsm} \) for multiplication dots. There is also a \( \texttt{\textbackslash dotsi} \) for dots with integrals as in

\[
\int_{A_1} \int_{A_2} \cdots \int_{A_n} f
\]

VIII.4.3. Delimiters

How do we produce something like

\[
\begin{vmatrix}
  a & h & g \\
  h & b & f \\
  g & f & c
\end{vmatrix} = 0
\]

Since \( \begin{vmatrix} \begin{smallmatrix} a & h & g \\ h & b & f \\ g & f & c \end{smallmatrix} \end{vmatrix} = 0 \), the matrix \( \begin{smallmatrix} a & h & g \\ h & b & f \\ g & f & c \end{smallmatrix} \) is not invertible.

Here the ‘small’ in-text matrices are produced by the environment \texttt{smallmatrix}. This environment \textit{does not} provide the enclosing delimiters \( ( ) \) or \( — — \) which we must supply as in

\[
\left| \begin{smallmatrix}
  a & h & g \\
  h & b & f \\
  g & f & c
\end{smallmatrix} \right| = 0
\]

the matrix \( \begin{smallmatrix} a & h & g \\ h & b & f \\ g & f & c \end{smallmatrix} \) is not invertible.

Why the \texttt{\textbackslash left}...\texttt{\textbackslash right} and \texttt{\textbackslash left}...\texttt{\textbackslash right}? These commands \texttt{\textbackslash left} and \texttt{\textbackslash right} enlarge the delimeter following them to the size of the enclosed material. To see their effect, try typesetting the above example without these commands. The list of symbols at the end of the chapter gives a list of delimiters that are available off the shelf.

One interesting point about the \texttt{\textbackslash left} and \texttt{\textbackslash right} pair is that, though every \texttt{\textbackslash left} should be matched to a \texttt{\textbackslash right}, the delimiters to which they apply need not match. In particular we can produce a single large delimiter produced by \texttt{\textbackslash left} or \texttt{\textbackslash right} by matching it with a matching command followed by a period. For example,

\[
\begin{cases}
  u_x = v_y \\
  u_y = -v_x
\end{cases}
\]

is produced by

\[
\left\{ \begin{array}{l}
  u_x = v_y \\
  u_y = -v_x
\end{array} \right\}
\]

Cauchy-Riemann Equations
There are instances where the delimiters produced by \( \left \) and \( \right \) are too small or too large. For example,

\[
\begin{equation*}
(x+y)^2 - (x-y)^2 = \left((x+y) + (x-y)\right)\left((x+y) - (x-y)\right) = 4xy
\end{equation*}
\]

gives

\[
(x + y)^2 - (x - y)^2 = ((x + y) + (x - y))((x + y) - (x - y)) = 4xy
\]

where the parentheses are all of the same size. But it may be better to make the outer ones a little larger to make the nesting visually apparent, as in

\[
\begin{equation*}
(x+y)^2 - (x-y)^2 = \left((x+y) + (x-y)\right)\left((x+y) - (x-y)\right) = 4xy
\end{equation*}
\]

gives

\[
(x + y)^2 - (x - y)^2 = \left((x + y) + (x - y)\right)\left((x + y) - (x - y)\right) = 4xy
\]

This is produced using the commands \( \bigl \) and \( \bigr \) before the outer parentheses as shown below:

\[
\begin{equation*}
(x+y)^2 - (x-y)^2 = \bigl((x+y) + (x-y)\bigr)\bigl((x+y) - (x-y)\bigr) = 4xy
\end{equation*}
\]

Apart from \( \bigl \) and \( \bigr \) there are \( \Bigl \), \( \biggl \) and \( \Biggl \) commands (and their \( r \) counterparts) which (in order) produce delimiters of increasing size. (Experiment with them to get a feel for their sizes.)

As another example, look at

\[
\begin{equation*}
For \( n \)-tuples of complex numbers \((x_1, x_2, \ldots, x_n)\) and \((y_1, y_2, \ldots, y_n)\) of complex numbers
\end{equation*}
\]

\[
\left(\sum_{k=1}^n |x_ky_k|\right)^2 \leq \left(\sum_{k=1}^n |x_k|\right)\left(\sum_{k=1}^n |y_k|\right)
\]

which is produced by

\[
\begin{equation*}
For \( n \)-tuples of complex numbers \((x_1, x_2, \ldots, x_n)\) and \((y_1, y_2, \ldots, y_n)\) of complex numbers
\end{equation*}
\]

\[
\begin{equation*}
\left(\sum_{k=1}^n |x_ky_k|\right)^2 \leq \left(\sum_{k=1}^n |x_k|\right)\left(\sum_{k=1}^n |y_k|\right)
\end{equation*}
\]

Does not the output below look better?
For $n$-tuples of complex numbers $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ of complex numbers

\[
\left( \sum_{k=1}^{n} |x_k y_k| \right)^2 \leq \left( \sum_{k=1}^{n} |x_k| \right) \left( \sum_{k=1}^{n} |y_k| \right)
\]

This one is produced by

\begin{verbatim}
For $n$-tuples of complex numbers $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ of complex numbers
\begin{equation*}
\left( \sum_{k=1}^{n} |x_k y_k| \right)^2 \leq \left( \sum_{k=1}^{n} |x_k| \right) \left( \sum_{k=1}^{n} |y_k| \right)
\end{equation*}
\end{verbatim}

Here the trouble is that the delimiters produced by \left and \right are a bit too large.

**VIII.4.4. Putting one over another**

Look at the following text

From the binomial theorem, it easily follows that if $n$ is an even number, then

\[
1 - \binom{n}{1} \frac{1}{2} + \binom{n}{2} \frac{1}{2^2} - \cdots - \binom{n}{n-1} \frac{1}{2^{n-1}} = 0
\]

We have fractions like \(\frac{n}{2n-1}\) and binomial coefficients like \(\binom{n}{2}\) here and the common feature of both is that they have one mathematical expression over another.

Fractions are produced by the \frac command which takes two arguments, the numerator followed by the denominator and the binomial coefficients are produced by the \binom command which also takes two arguments, the ‘top’ expression followed by the ‘bottom’ one. Thus the the input for the above example is

\begin{verbatim}
From the binomial theorem, it easily follows that if $n$ is an even number, then
\begin{equation*}
1 - \binom{n}{1} \frac{1}{2} + \binom{n}{2} \frac{1}{2^2} - \cdots - \binom{n}{n-1} \frac{1}{2^{n-1}} = 0
\end{equation*}
\end{verbatim}

You can see from the first paragraph above that the size of the outputs of \frac and \binom are smaller in text than in display. This default behavior has to be modified sometimes for nicer looking output. For example, consider the following output

Since \((x_n)\) converges to 0, there exists a positive integer $p$ such that

\[
|x_n| < \frac{1}{2} \quad \text{for all } n \geq p
\]

Would not it be nicer to make the fraction smaller and typeset this as

Since \((x_n)\) converges to 0, there exists a positive integer $p$ such that

\[
|x_n| < \frac{1}{2} \quad \text{for all } n \geq p
\]

The second output is produced by the input
Since $(x_n)$ converges to $0$, there exists a positive integer $p$ such that
\begin{equation*}
|x_n| < \frac{1}{2} \quad \text{for all } n \geq p
\end{equation*}

Note the use of the command \tfrac to produce a smaller fraction. (The first output is produced by the usual \frac command.)

There is also command \dfrac to produce a display style (larger size) fraction in text. Thus the sentence after the first example in this (sub)section can be typeset as

We have fractions like $\frac{1}{2}$ and ... 

by the input

We have fractions like $\dfrac{1}{2}$ and ...

As can be guessed, the original output was produced by \frac. Similarly, there are commands \dbinom (to produce display style binomial coefficients) and \tbinom (to produce text style binomial coefficients).

There is also a \genfrac command which can be used to produce custom fractions. To use it, we will have to specify six things

1. The left delimiter to be used—note that \{ must be specified as \\{
2. The right delimiter—again, \} to be specified as \}
3. The thickness of the horizontal line between the top expression and the bottom expression. If it is not specified, then it defaults to the ‘normal’ thickness. If it is set as 0pt then there will be no such line at all in the output.
4. The size of the output—this is specified as an integer 0, 1, 2 or 3, greater values corresponding to smaller sizes. (Technically these values correspond to \displaystyle, \textstyle, \scriptstyle and \scriptscriptstyle.)
5. The top expression
6. The bottom expression

Thus instead of \tfrac{1}{2}(2) we can also use \genfrac{}{}{}{1}{1}{2} and instead of \dbinom{n}{r}, we can also use \genfrac{(}{)}{0pt}{0}{1}{2} (but there is hardly any reason for doing so). More seriously, suppose we want to produce \genfrac{[}{]}{0pt}{0}{1}{2} as in

The Christoffel symbol [\gamma_{ij}^k] of the second kind is related to the Christoffel symbol [\gamma_{ij}] of the first kind by the equation

\begin{equation*}
[\gamma_{ij}^k] = g^{k1}[\gamma_{ij}^1] + g^{k2}[\gamma_{ij}^2]
\end{equation*}

This can be done by the input

The Christoffel symbol \[\gamma_{ij}^k\] of the second kind is related to the Christoffel symbol \[\gamma_{ij}\] of the first kind by the equation

\begin{equation*}
[\gamma_{ij}^k] = g^{k1}[\gamma_{ij}^1] + g^{k2}[\gamma_{ij}^2]
\end{equation*}

If such expressions are frequent in the document, it would be better to define ‘newcommand’ for them and use them instead of \genfrac every time as in the following input (which produces the same output as above).
The Christoffel symbol $\genfrac{[}{]}{0pt}{}{ij}{k}$ of the second kind is related to the Christoffel symbol $\genfrac{[}{]}{0pt}{}{ij}{1}$ of the first kind by the equation
\begin{equation*}
\genfrac{\{}{\}}{0pt}{}{ij}{k}=g^{k1}\genfrac{[}{]}{0pt}{}{ij}{1}+g^{k2}\genfrac{[}{]}{0pt}{}{ij}{2}
\end{equation*}

While on the topic of fractions, we should also mention the \cfrac command used to typeset continued fractions. For example, to get
\[\frac{4}{\pi}=1+\frac{1^2}{2+\cfrac{3^2}{2+\cfrac{5^2}{2+\ldots}}}\]
simply type
\begin{verbatim}
\begin{equation*}
\frac{4}{\pi}=1+\frac{1^2}{2+\frac{3^2}{2+\frac{5^2}{2+\frac{\ldots}{}}}}
\end{equation*}
\end{verbatim}

Some mathematicians would like to write the above equation as
\[\frac{4}{\pi}=1+\frac{1^2}{2}+\frac{3^2}{2}+\frac{5^2}{2}+\ldots\]

There is no ready-to-use command to produce this, but we can define one as follows
\begin{verbatim}
\newcommand{\cfplus}{\mathbin{\genfrac{}{}{0pt}{}{}{+}}}
\begin{equation*}
\frac{4}{\pi}=1+\frac{1^2}{2}\cfplus\frac{3^2}{2}\cfplus\frac{5^2}{2}\cfplus\ldots
\end{equation*}
\end{verbatim}

**VIII.4.5. Affixing symbols—over or under**

The table at the end of this chapter gives various math mode *accents* such as $\hat{a}$ and $\dot{a}$ to produce $\hat{a}$ and $\dot{a}$. But what if one needs $\check{a}$ or $\breve{a}$? The commands \overset and \underset come to the rescue. Thus $\overset{\circ}{a}$ produces $\check{a}$ and $\underset{\circ}{a}$ produces $\breve{a}$.

Basic \LaTeX provides the commands \xrightarrow and \xleftarrow also to put (extensible) arrows over symbols, as can be seen from the table. The amsmath package also provides the commands \overrightarrow and \overleftarrow to put (extensible) arrows below mathematical expressions.

Speaking of arrows, amsmath provides the commands \xrightarrow and \xleftarrow which produces arrows which can accommodate long texts as superscripts or subscripts. Thus we can produce
\[\overset{\text{long text}}{a}\]
Thus we see that

\[ 0 \rightarrow A \xrightarrow{f} B \xrightarrow{g} C \rightarrow 0 \]

is a short exact sequence

from the input

\begin{equation*}
0 \xrightarrow{} A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{} 0
\end{equation*}

is a short exact sequence

Note how the mandatory arguments of the first and last arrows are left empty to produce arrows with no superscripts. These commands also allow an optional argument (to be typed inside square brackets), which can be used to produce subscripts. For example

Thus we get

\begin{equation*}
0 \xrightarrow{} A \xrightarrow{\text{monic}} f \xrightarrow{} B \xrightarrow{\text{epi}} g \xrightarrow{} C \xrightarrow{} 0
\end{equation*}

gives

Thus we get

\[ 0 \rightarrow A \xrightarrow{\text{monic}} B \xrightarrow{\text{epi}} C \rightarrow 0 \]

By the way, would not it be nicer to make the two middle arrows the same width? This can be done by changing the command for the third arrow (the one from \(B\)) as shown below

Thus we get

\begin{equation*}
0 \xrightarrow{} A \xrightarrow{\text{monic}} f \xrightarrow{\text{epi}} B \xrightarrow{\text{epi}} g \xrightarrow{} C \xrightarrow{} 0
\end{equation*}

This gives

Thus we get

\[ 0 \rightarrow A \xrightarrow{\text{monic}} B \xrightarrow{\text{epi}} C \rightarrow 0 \]

where the lengths of the two arrows are almost the same. There are indeed ways to make the lengths exactly the same, but we will talk about it in another chapter.

Mathematical symbols are also attached as limits to such large operators as sum (\(\sum\)), product (\(\prod\)) set union (\(\cup\)), set intersection (\(\cap\)) and so on. The limits are input as subscripts or superscripts, but their positioning in the output is different in text and display. For example, the input
Euler not only proved that the series
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} \]
converges, but also that
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \]
gives the output

Euler not only proved that the series \( \sum_{n=1}^{\infty} \frac{1}{n^2} \) converges, but also that
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \]

Note that in display, the sum symbol is larger and the limits are put at the bottom and top (instead of at the sides, which is usually the case for subscripts and superscripts). If you want the same type of symbol (size, limits and all) in text also, simply change the line
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} \]
to
\[ \textstyle \sum_{n=1}^{\infty} \frac{1}{n^2} \]
and you will get

Euler not only proved that the series \( \sum_{n=1}^{\infty} \frac{1}{n^2} \) converges, but also that
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \]

(Note that this also changes the size of the fraction. What would you do to keep it small?) On the other hand, to make the displayed operator the same as in the text, add the command \textstyle before the \sum within the equation.

What if you only want to change the position of the limits but not the size of the operator in text? Then change the command
\[ \sum_{n=1}^{\infty} \frac{1}{n^2} \]
to
\[ \sum_{\textstyle n=1}^{\infty} \frac{1}{n^2} \]
and this will produce the output given below.

Euler not only proved that the series \( \sum_{n=1}^{\infty} \frac{1}{n^2} \) converges, but also that
\[ \sum_{\textstyle n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \]

On the other hand, if you want side-set limits in display type \texttt{nolimits} after the \sum within the equation as in
\[ \sum_{\text{nolimits n=1}}^{\infty} \frac{1}{n^2} \]
which gives
Euler not only proved that the series $\sum_{n=1}^{\infty} \frac{1}{n^2}$ converges, but also that
\[
\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}
\]

All these are true for other operators classified as “Variable-sized symbols”, except integrals. Though the integral symbol in display is larger, the position of the limits in both text and display is on the side as can be seen from the output below.

Thus
\[
\lim_{x \to \infty} \int_0^x \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
and so by definition,
\[
\int_0^\infty \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
which is produced by

\[
\lim_{x \to \infty} \int_0^x \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
and so by definition,
\[
\int_0^\infty \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
If you want the limits to be above and below the integral sign, just add the command \limits immediately after the \int command. Thus

Thus
\[
\lim_{x \to \infty} \int_0^x \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
and so by definition,
\[
\int_0^\infty \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
gives

Thus
\[
\lim_{x \to \infty} \int_0^x \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
and so by definition,
\[
\int_0^\infty \frac{\sin x}{x} \, \mathrm{d}x = \frac{\pi}{2}
\]
Now how do we typeset something like
\[
p_k(x) = \prod_{i=1}^{n} \frac{x-f_i}{l_i-x}
\]
where we have two lines of subscripts for \prod? There is a command \substack which will do the trick. The above output is obtained from
\begin{equation*}
p_k(x) = \prod_{\substack{i=1\atop i \neq k}}^n \left( \frac{x-t_i}{t_k-t_i} \right)
\end{equation*}

The amsmath package has also a \texttt{sideset} command which can be used to put symbols at any of the four corners of a large operator. Thus
\[\sideset{_{ll}^{ul}}{_{lr}^{ur}}\bigcup\]
produces \( ul \bigcup ur \)
\[\sideset{}{'}\sum\]
produces \( \sum' \).

\section{New Operators}

Mathematical text is usually typeset in italics, and \TeX{} follows this tradition. But certain functions in mathematics such as \texttt{log}, \texttt{sin}, \texttt{lim} and so on are traditionally typeset in roman. This is implemented in \TeX{} by the use of commands like \texttt{\log}, \texttt{\sin}, \texttt{\lim} and so on. The symbols classified as “Log-like symbols” in the table at the end of this chapter shows such functions which are predefined in \LaTeX{}.

Having read thus far, it may be no surprise to learn that we can define our own “operator names” which receive this special typographic treatment. This is done by the \texttt{\DeclareMathOperator} command. Thus if the operator \texttt{cl} occurs frequently in the document, you can make the declaration

\begin{verbatim}
\DeclareMathOperator{\cl}{cl}
\end{verbatim}

in the preamble and then type \texttt{\cl(A)} to produce \( \text{cl}(A) \), for example.

Note that an operator defined like this accommodates subscripts and superscripts in the usual way, that is, at its sides. Thus

We denote the closure of \( A \) in the subspace \( Y \) of \( X \) by

\[\cl_Y(A)\]

produces

We denote the closure of \( A \) in the subspace \( Y \) of \( X \) by \( \text{cl}_Y(A) \).

If we want to define a new operator with subscripts and superscripts placed in the “limits” position below and above, then we should use the starred form of the \texttt{\DeclareMathOperator} as shown below

\begin{verbatim}
\DeclareMathOperator*{\esup}{ess\,sup}
\end{verbatim}

For \( f \in L^\infty(R) \), we define
\begin{equation*}
||f||_{L^\infty} = \esup_{x \in R} |f(x)|
\end{equation*}

(Note that the declaration \texttt{must} be done in the preamble.) This produces the output

For \( f \in L^\infty(R) \), we define

\[ ||f||_{L^\infty} = \esup_{x \in R} |f(x)| \]

(Why the \texttt{,} command in the definition?)
viii.6. The Many Faces of Mathematics

We have noted that most mathematics is typeset in italics typeface and some mathematical operators are typeset in an upright fashion. There may be need for additional typefaces as in typesetting vectors in boldface.

\TeX{} includes several styles to typeset mathematics as shown in the table below.

<table>
<thead>
<tr>
<th>TYPE STYLE</th>
<th>COMMAND</th>
<th>EXAMPLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>italic</td>
<td>\textit</td>
<td>$x+y=z$</td>
</tr>
<tr>
<td>roman</td>
<td>\texttt</td>
<td>\texttt{x+y=z}</td>
</tr>
<tr>
<td>bold</td>
<td>\textbf</td>
<td>\textbf{x+y=z}</td>
</tr>
<tr>
<td>sans serif</td>
<td>\textsf</td>
<td>\textsf{x+y=z}</td>
</tr>
<tr>
<td>typewriter</td>
<td>\texttt</td>
<td>\texttt{x+y=z}</td>
</tr>
<tr>
<td>calligraphic</td>
<td>\textsc</td>
<td>\textsc{x+y=z}</td>
</tr>
<tr>
<td>(upper case only)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In addition to these, several other math alphabets are available in various packages (some of which are shown in the list of symbols at the end of this chapter).

Note that the command \textbf produces only roman boldface and not math italic boldface. Sometimes you may need boldface math italic, for example to typeset vectors. For this, amsmath provides the \textbf command. Thus we can get

\begin{equation*}
\textbf{a+b=c}
\end{equation*}

If the document contains several occurrences of such symbols, it is better to make a new definition such as

\newcommand{\vect}[1]{\textbf{#1}}

and then use $\vect{a}$ to produce a and $\vect{b}$ to produce b and so on. the additional advantage of this approach is that if you change your mind later and want vectors to be typeset with arrows above them as $\vec{a}$, then all you need is to change the \textbf part of the definition of \vect to \overrightarrow and the change will be effected throughout the document.

Now if we change the input of the above example as

\begin{equation*}
\textbf{a+b=c}
\end{equation*}

then we get the output
In this case, we define

\[ a + b = c \]

Note that now the symbols + and = are also in boldface. Thus `\boldsymbol` makes bold every math symbol in its scope (provided the bold version of that symbol is available in the current math font).

There is another reason for tweaking the math fonts. Recently, the International Standards Organization (ISO) has established the recognized typesetting standards in mathematics. Some of the points in it are,

1. Simple variables are represented by italic letters as \( a, x \).
2. Vectors are written in boldface italic as \( \mathbf{a}, \mathbf{x} \).
3. Matrices may appear in sans serif as in \( \mathsf{A}, \mathsf{X} \).
4. The special numbers \( e, \imath \) and the differential operator \( d \) are written in upright roman.

Point 1 is the default in \LaTeX{} and we have seen how point 2 can be implemented. To fulfill Point 4, it is enough if we define something like

\begin{verbatim}
\newcommand{\me}{\mathrm{e}}
\newcommand{\mi}{\mathrm{i}}
\newcommand{\diff}{\mathrm{d}}
\end{verbatim}

and then use \$\me\$ for \( e \) and \$\mi\$ for \( i \) and \$\diff x\$ for \( dx \).

Point 3 can be implemented using \texttt{\mathsf} but it is a bit difficult (but not impossible) if we need them to be in italic also. The solution is to create a new math alphabet, say, \texttt{\mathsfsl} by the command

\begin{verbatim}
\DeclareMathAlphabet{\mathsfsl}{OT1}{cmss}{m}{sl}
\end{verbatim}

(in the preamble) and use it to define a command \texttt{\matr} to typeset matrices in this font by

\begin{verbatim}
\newcommand{\matr}[1]{\ensuremath{\mathsfsl{#1}}}
\end{verbatim}

so that \$\matr A\$ produces \( A \).

\section{And that is not all!}

We have only briefly discussed the basic techniques of typesetting mathematics using \LaTeX{} and some of the features of the \texttt{amsmath} package which helps us in this task. For more details on this package see the document \texttt{amsldoc.dvi} which should be available with your \TeX{} distribution. If you want to produce \textit{really} beautiful mathematical documents, read the Master—"The \TeX{} Book" by Donald Knuth, especially Chapter 18, "Fine Points of Mathematics Typing".

\section{Symbols}

Table VIII.1: Greek Letters

\begin{tabular}{|c|c|c|c|c|c|c|c|}
\hline
\( \alpha \) & \( \beta \) & \( \gamma \) & \( \delta \) & \( \epsilon \) & \( \zeta \) & \( \eta \) & \( \theta \) \\
\hline
\( \\alpha \) & \( \\beta \) & \( \\gamma \) & \( \\delta \) & \( \\epsilon \) & \( \\zeta \) & \( \\eta \) & \( \\theta \) \\
\hline
\end{tabular}
Table VIII.2: Binary Operation Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>±</td>
<td>\pm</td>
</tr>
<tr>
<td>⫷</td>
<td>\mp</td>
</tr>
<tr>
<td>∩</td>
<td>\cap</td>
</tr>
<tr>
<td>◦</td>
<td>\diamond</td>
</tr>
<tr>
<td>⊕</td>
<td>\oplus</td>
</tr>
<tr>
<td>☓</td>
<td>\mpm</td>
</tr>
<tr>
<td>⊖</td>
<td>\oslash</td>
</tr>
<tr>
<td>×</td>
<td>\times</td>
</tr>
<tr>
<td>⊕</td>
<td>\uplus</td>
</tr>
<tr>
<td>△</td>
<td>\bigtriangleup</td>
</tr>
<tr>
<td>⊖</td>
<td>\otimes</td>
</tr>
<tr>
<td>÷</td>
<td>\div</td>
</tr>
<tr>
<td>⊕</td>
<td>\bigotimes</td>
</tr>
<tr>
<td>⊕</td>
<td>\odot</td>
</tr>
<tr>
<td>⫷</td>
<td>\sqcup</td>
</tr>
<tr>
<td>⫷</td>
<td>\sqcap</td>
</tr>
<tr>
<td>⊕</td>
<td>\triangleleft</td>
</tr>
<tr>
<td>⊕</td>
<td>\bigcirc</td>
</tr>
<tr>
<td>◦</td>
<td>\circ</td>
</tr>
<tr>
<td>⊕</td>
<td>\biguplus</td>
</tr>
</tbody>
</table>

* Not predefined in \LaTeX. Use one of the packages latexsym, amsfonts or amssymb.

Table VIII.3: Relation Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>≤</td>
<td>\leq</td>
</tr>
<tr>
<td>≥</td>
<td>\geq</td>
</tr>
<tr>
<td>≡</td>
<td>\equiv</td>
</tr>
<tr>
<td>⊢</td>
<td>\models</td>
</tr>
<tr>
<td>≣</td>
<td>\gg</td>
</tr>
<tr>
<td>≦</td>
<td>\succeq</td>
</tr>
<tr>
<td>≦</td>
<td>\preceq</td>
</tr>
<tr>
<td>⊆</td>
<td>\subseteq</td>
</tr>
<tr>
<td>⊇</td>
<td>\supset</td>
</tr>
<tr>
<td>≠</td>
<td>\neq</td>
</tr>
<tr>
<td>⊂</td>
<td>\subset</td>
</tr>
<tr>
<td>⊃</td>
<td>\supset</td>
</tr>
<tr>
<td>⊆</td>
<td>\sqsubseteq</td>
</tr>
<tr>
<td>⊇</td>
<td>\sqsupset</td>
</tr>
<tr>
<td>⊆</td>
<td>\setminus</td>
</tr>
<tr>
<td>⊇</td>
<td>\approx</td>
</tr>
<tr>
<td>≍</td>
<td>\asymp</td>
</tr>
<tr>
<td>⊏</td>
<td>\sqsubset</td>
</tr>
<tr>
<td>⊐</td>
<td>\sqsupset</td>
</tr>
<tr>
<td>⊏</td>
<td>\propto</td>
</tr>
<tr>
<td>⊐</td>
<td>\pivot</td>
</tr>
<tr>
<td>⊏</td>
<td>\dagger</td>
</tr>
<tr>
<td>⊐</td>
<td>\ddagger</td>
</tr>
<tr>
<td>⊖</td>
<td>\Join</td>
</tr>
<tr>
<td>⊖</td>
<td>\rightleftharpoons</td>
</tr>
<tr>
<td>⊖</td>
<td>\bowtie</td>
</tr>
<tr>
<td>⊖</td>
<td>\Join</td>
</tr>
<tr>
<td>⊖</td>
<td>\bowtie</td>
</tr>
<tr>
<td>⊖</td>
<td>\Join</td>
</tr>
<tr>
<td>⊖</td>
<td>\bowtie</td>
</tr>
</tbody>
</table>

* Not predefined in \LaTeX. Use one of the packages latexsym, amsfonts or amssymb.

Table VIII.4: Punctuation Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>,</td>
<td>,</td>
</tr>
<tr>
<td>;</td>
<td>;</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
</tbody>
</table>

Table VIII.5: Arrow Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>←</td>
<td>\leftarrow</td>
</tr>
<tr>
<td>⇐</td>
<td>\Leftarrow</td>
</tr>
<tr>
<td>→</td>
<td>\rightarrow</td>
</tr>
<tr>
<td>⇒</td>
<td>\Rightarrow</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
<tr>
<td>⇔</td>
<td>\iff</td>
</tr>
</tbody>
</table>

* Not predefined in \LaTeX. Use one of the packages latexsym, amsfonts or amssymb.
### VIII.8. Symbols

<table>
<thead>
<tr>
<th>Symbols</th>
<th>\hookleftarrow</th>
<th>\hookrightarrow</th>
<th>\searrow</th>
</tr>
</thead>
<tbody>
<tr>
<td>\leftharpoonup</td>
<td>\leftharpoondown</td>
<td>\rightharpoonup</td>
<td>\rightharpoondown</td>
</tr>
<tr>
<td>\leadsto</td>
<td>\rightarrow</td>
<td>\leftarrow</td>
<td>\leftrightarrow</td>
</tr>
</tbody>
</table>

* Not predefined in \LaTeX. Use one of the packages latexsym, amsfonts or amssymb.

#### Table VIII.6: Miscellaneous Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ldots</td>
<td>Ellipsis</td>
</tr>
<tr>
<td>\aleph</td>
<td>Aleph</td>
</tr>
<tr>
<td>\prime</td>
<td>Prime</td>
</tr>
<tr>
<td>\forall</td>
<td>For all</td>
</tr>
<tr>
<td>\infty</td>
<td>Infinity</td>
</tr>
<tr>
<td>\hbar</td>
<td>H Bar</td>
</tr>
<tr>
<td>\emptyset</td>
<td>Empty set</td>
</tr>
<tr>
<td>\exists</td>
<td>Exists</td>
</tr>
<tr>
<td>\Box</td>
<td>Box</td>
</tr>
<tr>
<td>\nabla</td>
<td>Nabla</td>
</tr>
<tr>
<td>\top</td>
<td>Top</td>
</tr>
<tr>
<td>\natural</td>
<td>Natural</td>
</tr>
<tr>
<td>\flat</td>
<td>Flat</td>
</tr>
<tr>
<td>\daleth</td>
<td>Daleth</td>
</tr>
<tr>
<td>\wp</td>
<td>Wp</td>
</tr>
<tr>
<td>\bot</td>
<td>Bot</td>
</tr>
<tr>
<td>\sharp</td>
<td>Sharp</td>
</tr>
<tr>
<td>\Re</td>
<td>Re</td>
</tr>
<tr>
<td>\top</td>
<td>Top</td>
</tr>
<tr>
<td>\clubsuit</td>
<td>Clubsuit</td>
</tr>
<tr>
<td>\heartsuit</td>
<td>Heartsuit</td>
</tr>
<tr>
<td>\natural</td>
<td>Natural</td>
</tr>
<tr>
<td>\partial</td>
<td>Partial</td>
</tr>
<tr>
<td>\diamondsuit</td>
<td>Diamondsuit</td>
</tr>
<tr>
<td>\imath</td>
<td>Imath</td>
</tr>
<tr>
<td>\nabla</td>
<td>Nabla</td>
</tr>
<tr>
<td>\top</td>
<td>Top</td>
</tr>
<tr>
<td>\natural</td>
<td>Natural</td>
</tr>
<tr>
<td>\flat</td>
<td>Flat</td>
</tr>
<tr>
<td>\daleth</td>
<td>Daleth</td>
</tr>
<tr>
<td>\wp</td>
<td>Wp</td>
</tr>
<tr>
<td>\bot</td>
<td>Bot</td>
</tr>
<tr>
<td>\sharp</td>
<td>Sharp</td>
</tr>
<tr>
<td>\Re</td>
<td>Re</td>
</tr>
<tr>
<td>\top</td>
<td>Top</td>
</tr>
<tr>
<td>\clubsuit</td>
<td>Clubsuit</td>
</tr>
<tr>
<td>\heartsuit</td>
<td>Heartsuit</td>
</tr>
<tr>
<td>\natural</td>
<td>Natural</td>
</tr>
<tr>
<td>\partial</td>
<td>Partial</td>
</tr>
<tr>
<td>\diamondsuit</td>
<td>Diamondsuit</td>
</tr>
<tr>
<td>\imath</td>
<td>Imath</td>
</tr>
</tbody>
</table>

* Not predefined in \LaTeX. Use one of the packages latexsym, amsfonts or amssymb.

#### Table VIII.7: Variable-sized Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>\sum</td>
<td>Sum</td>
</tr>
<tr>
<td>\prod</td>
<td>Product</td>
</tr>
<tr>
<td>\coprod</td>
<td>Coprod</td>
</tr>
<tr>
<td>\int</td>
<td>Integral</td>
</tr>
<tr>
<td>\oint</td>
<td>Contour integral</td>
</tr>
</tbody>
</table>

#### Table VIII.8: Log-like Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>\arccos</td>
<td>Arccos</td>
</tr>
<tr>
<td>\cos</td>
<td>Cos</td>
</tr>
<tr>
<td>\csc</td>
<td>Csc</td>
</tr>
<tr>
<td>\exp</td>
<td>Exp</td>
</tr>
<tr>
<td>\ker</td>
<td>Ker</td>
</tr>
<tr>
<td>\limsup</td>
<td>Limsup</td>
</tr>
<tr>
<td>\min</td>
<td>Min</td>
</tr>
<tr>
<td>\sinh</td>
<td>Sinh</td>
</tr>
<tr>
<td>\arcsin</td>
<td>Arcsin</td>
</tr>
<tr>
<td>\cosh</td>
<td>Cosh</td>
</tr>
<tr>
<td>\deg</td>
<td>Deg</td>
</tr>
<tr>
<td>\gcd</td>
<td>Gcd</td>
</tr>
<tr>
<td>\lg</td>
<td>Lg</td>
</tr>
<tr>
<td>\ln</td>
<td>Ln</td>
</tr>
<tr>
<td>\Pr</td>
<td>Pr</td>
</tr>
<tr>
<td>\sup</td>
<td>Sup</td>
</tr>
<tr>
<td>\arctan</td>
<td>Arctan</td>
</tr>
<tr>
<td>\cot</td>
<td>Cot</td>
</tr>
<tr>
<td>\det</td>
<td>Det</td>
</tr>
<tr>
<td>\hom</td>
<td>Hom</td>
</tr>
<tr>
<td>\lim</td>
<td>Lim</td>
</tr>
<tr>
<td>\log</td>
<td>Log</td>
</tr>
<tr>
<td>\sec</td>
<td>Sec</td>
</tr>
<tr>
<td>\tan</td>
<td>Tan</td>
</tr>
<tr>
<td>\arg</td>
<td>Arg</td>
</tr>
<tr>
<td>\coth</td>
<td>Coth</td>
</tr>
<tr>
<td>\dim</td>
<td>Dim</td>
</tr>
<tr>
<td>\inf</td>
<td>Inf</td>
</tr>
<tr>
<td>\liminf</td>
<td>Liminf</td>
</tr>
<tr>
<td>\max</td>
<td>Max</td>
</tr>
<tr>
<td>\sin</td>
<td>Sin</td>
</tr>
<tr>
<td>\tanh</td>
<td>Tanh</td>
</tr>
</tbody>
</table>

#### Table VIII.9: Delimiters

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>)</td>
<td>)</td>
</tr>
<tr>
<td>[</td>
<td>[</td>
</tr>
<tr>
<td>]</td>
<td>]</td>
</tr>
<tr>
<td>{</td>
<td>{</td>
</tr>
<tr>
<td>}</td>
<td>}</td>
</tr>
<tr>
<td>\lfloor</td>
<td>\lfloor</td>
</tr>
<tr>
<td>\rfloor</td>
<td>\rfloor</td>
</tr>
<tr>
<td>\langle</td>
<td>\langle</td>
</tr>
<tr>
<td>\rangle</td>
<td>\rangle</td>
</tr>
<tr>
<td>\lceil</td>
<td>\lceil</td>
</tr>
<tr>
<td>\rceil</td>
<td>\rceil</td>
</tr>
<tr>
<td>\backslash</td>
<td>\backslash</td>
</tr>
</tbody>
</table>

#### Table VIII.10: Large Delimiters

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>\lgroup</td>
<td>Lgroup</td>
</tr>
<tr>
<td>\rgroup</td>
<td>Rgroup</td>
</tr>
<tr>
<td>\arrowvert</td>
<td>Arrowvert</td>
</tr>
<tr>
<td>\bracevert</td>
<td>Bracevert</td>
</tr>
</tbody>
</table>
Table VIII.11: Math Mode Accents
\[ \hat{a} \quad \acute{a} \quad \bar{a} \quad \dot{a} \quad \breve{a} \quad \check{a} \quad \grave{a} \quad \vec{a} \quad \ddot{a} \quad \tilde{a} \]

Table VIII.12: Some other Constructions
\[ \overleftarrow{abc} \quad \overrightarrow{abc} \quad \overline{abc} \quad \underline{abc} \quad \overbrace{abc} \quad \underbrace{abc} \quad \sqrt{abc} \quad \sqrt[n]{abc} \quad f' \quad \frac{abc}{xyz} \]

Table VIII.13: AMS Delimiters
\[ \ulcorner \quad \urcorner \quad \llcorner \quad \lrcorner \]

Table VIII.14: AMS Arrows
\[ \dashrightarrow \quad \dashleftarrow \quad \leftleftarrows \quad \leftrightharpoons \quad \curvearrowleft \quad \circlearrowleft \quad \Lleftarrow \quad \Lsh \quad \uparrow \quad \upharpoonleft \quad \downarrow \quad \downharpoonleft \quad \rightsquigarrow \quad \leftrightsquigarrow \quad \rightrightarrows \]

Table VIII.15: AMS Negated Arrows
\[ \nleftarrow \quad \nrightarrow \quad \nLeftarrow \quad \nRightarrow \quad \nleftrightarrow \quad \nLeftrightarrow \]

Table VIII.16: AMS Greek
## 8. Symbols

- \digamma \quad \varkappa

### Table VIII.17: AMS Hebrew

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ב</td>
<td>\beth</td>
</tr>
<tr>
<td>ה</td>
<td>\daleth</td>
</tr>
<tr>
<td>ג</td>
<td>\gimel</td>
</tr>
</tbody>
</table>

### Table VIII.18: AMS Miscellaneous

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ℬ</td>
<td>\beth</td>
</tr>
<tr>
<td>ℸ</td>
<td>\daleth</td>
</tr>
<tr>
<td>ג</td>
<td>\gimel</td>
</tr>
</tbody>
</table>

### Table VIII.19: AMS Binary Operators

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊕</td>
<td>\dotplus</td>
</tr>
<tr>
<td>\smallsetminus</td>
<td>\setminus</td>
</tr>
<tr>
<td>⋂</td>
<td>\cap</td>
</tr>
<tr>
<td>\barwedge</td>
<td>\veebar</td>
</tr>
<tr>
<td>⊙</td>
<td>\boxtimes</td>
</tr>
<tr>
<td>⋊</td>
<td>\rtimes</td>
</tr>
<tr>
<td>⋒</td>
<td>\Cap</td>
</tr>
<tr>
<td>\circleddash</td>
<td>\circledcirc</td>
</tr>
<tr>
<td>⊳</td>
<td>\triangledown</td>
</tr>
<tr>
<td>⩾</td>
<td>\geqslant</td>
</tr>
</tbody>
</table>

### Table VIII.20: AMS Binary Relations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>⪅</td>
<td>\lessapprox</td>
</tr>
<tr>
<td>⪆</td>
<td>\approxeq</td>
</tr>
<tr>
<td>⪇</td>
<td>\lesseqqgtr</td>
</tr>
<tr>
<td>≘</td>
<td>\fallingdotseq</td>
</tr>
<tr>
<td>⪊</td>
<td>\curlyeqprec</td>
</tr>
<tr>
<td>≐</td>
<td>\eqcirc</td>
</tr>
<tr>
<td>≖</td>
<td>\eqcirc</td>
</tr>
<tr>
<td>\gtrless</td>
<td>\asymp</td>
</tr>
<tr>
<td>\supseteqq</td>
<td>\nsupseteqq</td>
</tr>
<tr>
<td>\eqless</td>
<td>\lesseqqgtr</td>
</tr>
<tr>
<td>\approxeq</td>
<td>\approxeq</td>
</tr>
<tr>
<td>\lesseqqgtr</td>
<td>\lesseqqgtr</td>
</tr>
<tr>
<td>\fallingdotseq</td>
<td>\fallingdotseq</td>
</tr>
<tr>
<td>\Subset</td>
<td>\Subset</td>
</tr>
<tr>
<td>\circledcirc</td>
<td>\circledcirc</td>
</tr>
<tr>
<td>\triangledown</td>
<td>\triangledown</td>
</tr>
<tr>
<td>\supseteqq</td>
<td>\supseteqq</td>
</tr>
<tr>
<td>\asymp</td>
<td>\asymp</td>
</tr>
<tr>
<td>\nsupseteqq</td>
<td>\nsupseteqq</td>
</tr>
<tr>
<td>\eqcirc</td>
<td>\eqcirc</td>
</tr>
<tr>
<td>≖</td>
<td>\eqcirc</td>
</tr>
<tr>
<td>≘</td>
<td>\eqcirc</td>
</tr>
</tbody>
</table>

- \leqq \quad \leqslant \quad \leqslnless
Table VIII.21: AMS Negated Binary Relations

| \nless | \nleq | \nleqslant |
| \nleq | \nleq | \lvertneqq |
| \lnapprox | \nprec | \nprecneqq |
| \precnapprox | \nsim | \nshortmid |
| \nvDash | \nvDash | \ntriangleleft |
| \nsupseteqq | \nsupseteqq | \nsuccsim |
| \nshortparallel | \nparallel | \ntrianglelefteq |
| \ntrianglerighteq | \ntrianglelefteq | \subsetneqq |
| \ncong | \ncong | \ncong |
| \nvarsupseteqq | \nvarsupseteqq | \nbacksimeq |

Table VIII.22: Math Alphabets

| ABCdef | \mathbb{ABCdef} |
| ABCdef | \mathit{ABCdef} |
| ABCdef | \mathnormal{ABCdef} |
| ABC | \mathcal{ABC} |
| ABC | \mathcal{ABC} | euscript with option: mathcal |
| ABC | \mathcal{ABC} | euscript with option: mathfrak |
| ABC | \mathfrak{ABCdef} | eufrak |
| ABC | \mathbb{ABC} | amsfonts or amssymb |
| ABC | \mathscr{ABC} | mathsfs |
TUTORIAL IX

TYPESETTING THEOREMS

ix.1. THEOREMS IN \LaTeX

In Mathematical documents we often have special statements such as axioms (which are nothing but the assumptions made) and theorems (which are the conclusions obtained, sometimes known by other names like propositions or lemmas). These are often typeset in different font to distinguish them from surrounding text and given a name and a number for subsequent reference. Such distinguished statements are now increasingly seen in other subjects also. We use the term theorem-like statements for all such statements.

\LaTeX{} provides the declaration \texttt{\newtheorem} to define the theorem-like statements needed in a document. This command has two arguments, the first for the name we assign to the environment and the second, the name to be printed with the statement. Thus if you want

\begin{quote}
\textbf{Theorem 1.} The sum of the angles of a triangle is $180^\circ$.
\end{quote}

you first specify

\texttt{\newtheorem{thm}{Theorem}}

and then type

\begin{quote}
\begin{thm}
The sum of the angles of a triangle is $180^\circ$.
\end{thm}
\end{quote}

Note that in the command \texttt{\newtheorem} the first argument can be any name you fancy, instead of the \texttt{thm} given here. Also, it is a good idea to keep all your \texttt{\newtheorem} commands together in the preamble.

The \texttt{\newtheorem} command has a couple of optional arguments which control the way the corresponding statement is numbered. For example if you want the above theorem to be numbered \texttt{1.1} (the first theorem of the first section) rather than a plain 1, then you must specify

\texttt{\newtheorem{thm}{Theorem}[section]}

in the \texttt{\newtheorem} command. Then the same input as above for the theorem produces

\begin{quote}
\textbf{Theorem IX.1.1.} The sum of the angles of a triangle is $180^\circ$.
\end{quote}

The next \texttt{Theorem} will be numbered \texttt{1.2}, the third \texttt{Theorem} in the fourth section will be numbered \texttt{4.3} and so on.
The other optional argument of the `\newtheorem` command is useful when you have several different types of theorem-like statements (such as lemmas and corollaries) and you want some of them to share the same numbering sequence. For example, if you want

**Theorem IX.1.2.** The sum of the angles of a triangle is $180^\circ$.

An immediate consequence of the result is the following

**Corollary IX.1.3.** The sum of the angles of a quadrilateral is $360^\circ$.

Then you must specify

```
\newtheorem{cor}[thm]{Corollary}
```

after the specification `\newtheorem{thm}[section]` and then type

```
\begin{thm}
  The sum of the angles of a triangle is $180^\circ$.
\end{thm}
```

An immediate consequence of the result is the following

**Corollary IX.1.4.** The sum of the angles of a quadrilateral is $360^\circ$.

The optional argument `thm` in the definition of the `cor` environment specifies that “Corollaries” and “Theorems” are to be numbered in the same sequence.

A theorem-like environment defined using the `\newtheorem` command has also an optional argument which is used to give a note about the theorem such as the name of its discoverer or its own common name. For example, to get

**Theorem IX.1.5 (Euclid).** The sum of the angles of a triangle is $180^\circ$.

you must type

```
\begin{thm}[Euclid]
  The sum of the angles of a triangle is $180^\circ$.
\end{thm}
```

Note the optional argument `Euclid` after the `\begin{thm}`. This use of `[]` for optional notes sometimes lead to unintended results. For example, to get

**Theorem IX.1.6.** $[0,1]$ is a compact subset of $\mathbb{R}$.

if you type

```
\begin{thm}
  $[0,1]$ is a compact subset of $\mathbb{R}$.
\end{thm}
```

then you get

**Theorem IX.1.7 (0,1).** is a compact subset of $\mathbb{R}$.

Do you see what happened? The string $0,1$ within $[]$ at the beginning of the theorem is considered an optional note by \LaTeX! The correct way is to type
\begin{thm}
$[0,1]$ is a compact subset of $\mathbb{R}$.
\end{thm}

Now all the theorem-like statements produced above have the same typographical form—name and number in boldface and the body of the statement in italics. What if you need something like

\begin{center}
\textbf{Theorem IX.1.1 (Euclid).} The sum of the angles of a triangle is 180°. \\
\end{center}

Such customization is necessitated not only by the aesthetics of the author but often by the whims of the designers in publishing houses also.

### IX.2. Designer Theorems—The amsthm Package

The package \texttt{amsthm} affords a high level of customization in formatting theorem-like statements. Let us first look at the predefined styles available in this package.

#### IX.2.1. Ready made styles

The default style (this is what you get if you do not say anything about the style) is termed \texttt{plain} and it is what we have seen so far—name and number in boldface and body in italic. Then there is the definition style which gives name and number in boldface and body in roman. And finally there is the remark style which gives number and name in italics and body in roman.

For example if you put in the preamble
\begin{verbatim}
\usepackage{amsthm}
\newtheorem{thm}{Theorem}[section]
\theoremstyle{definition}
\newtheorem{dfn}{Definition}[section]
\theoremstyle{remark}
\newtheorem{note}{Note}[section]
\theoremstyle{plain}
\newtheorem{lem}[thm]{Lemma}
\end{verbatim}

and then type somewhere in your document
\begin{verbatim}
\begin{dfn}
A triangle is the figure formed by joining each pair of three non collinear points by line segments.
\end{dfn}

\begin{note}
A triangle has three angles.
\end{note}

\begin{thm}
The sum of the angles of a triangle is $180^\circ$.
\end{thm}

\begin{lem}
The sum of any two sides of a triangle is greater than or equal to the third.
\end{lem}
\end{verbatim}
then you get

<table>
<thead>
<tr>
<th>Definition IX.2.1. A triangle is the figure formed by joining each pair of three non collinear points by line segments.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note IX.2.1. A triangle has three angles.</td>
</tr>
<tr>
<td>Theorem IX.2.1. The sum of the angles of a triangle is 180°.</td>
</tr>
<tr>
<td>Lemma IX.2.2. The sum of any two sides of a triangle is greater than or equal to the third.</td>
</tr>
</tbody>
</table>

Note how the `\textit{remark}` command is used to switch between various styles, especially the last `\textit{remark}` command. Without it, the previous `\textit{remark}` will still be in force when `\em` is defined and so “Lemma” will be typeset in the `\textit{remark}` style.

**IX.2.2. Custom made theorems**

Now we are ready to roll our own “theorem styles”. This is done via the `\newtheoremstyle` command, which allows us to control almost all aspects of typesetting theorem like statements. This command has nine parameters and the general syntax is

```
\newtheoremstyle{% 
{name}% 
{abovespace}% 
{belowspace}% 
{bodyfont}% 
{indent}% 
{headfont}% 
{headpunct}% 
{headspace}% 
{custom-head-spec}% 
```

The first parameter `name` is the name of the new style. Note that it is *not* the name of the `\textit{environment}` which is to be used later. Thus in the example above `\textit{remark}` is the name of a new style for typesetting theorem like statements and `\textit{note}` is the name of the environment subsequently defined to have this style (and `\textit{note}` is the name of the statement itself).

The next two parameters determine the vertical space between the theorem and the surrounding text—the `abovespace` is the space from the preceding text and the `belowspace` the space from the following text. You can specify either a rigid length (such as `12pt`) or a rubber length (such as `\baselineskip`) as a value for either of these. Leaving either of these empty sets them to the “usual values” (Technically the `\topsep`).

The fourth parameter `bodyfont` specifies the font to be used for the body of the theorem-like statement. This is to be given as a `\textit{declaration}` such as `\textsc{scshape} or `\textbf{bfseries and not as a \textit{command}} such as `\texttt{texttt}` or `\texttt{textbf}. If this is left empty, then the main text font of the document is used.

The next four parameters refer to the `\textit{theorembhead}`—the part of the theorem like statement consisting of the name, number and the optional note. The fifth parameter `indent` specifies the indentation of `\textit{theorembhead} from the left margin. If this is empty, then there is no indentation of the `\textit{theorembhead} from the left margin. The next parameter specifies the font to be used for the `\textit{theorembhead}. The comments about the parameter `bodyfont, made in the previous paragraph holds for this also. The parameter `headpunct`
(the seventh in our list) is for specifying the punctuation after the theoremhead. If you do not want any, you can leave this empty. The last parameter in this category (the last but one in the entire list), namely headspace, determines the (horizontal) space to be left between the theoremhead and the theorembody. If you want only a normal interword space here put a single blank space as { } in this place. (Note that it is not the same as leaving this empty as in {}). Another option here is to put the command \newline here. Then instead of a space, you get a linebreak in the output; that is, the theoremhead will be printed in a line by itself and the theorembody starts from the next line.

The last parameter custom-head-spec is for customizing theoremheads. Since it needs some explanation (and since we are definitely in need of some breathing space), let us now look at a few examples using the eight parameters we’ve already discussed.

It is almost obvious now how the last theorem in Section 1 (see Page 111) was designed. It was generated by
\newtheoremstyle{mystyle}{}{}{\slshape}{}{\scshape}{.}{ }{}
\theoremstyle{mystyle}
\newtheorem{mythm}{Theorem}[section]
\begin{mythm}
The sum of the angles of a triangle is $180^\circ$.
\end{mythm}
As another example, consider the following
\newtheoremstyle{mynewstyle}{12pt}{12pt}{\itshape}{}{}{\sffamily}{:}{\newline}{}
\theoremstyle{mynewstyle}
\newtheorem{mynewthm}{Theorem}[section]
\begin{mynewthm}[Euclid]
The sum of the angles of a triangle is $180^\circ$.
\end{mynewthm}
This produces
Theorem IX.2.1 (Euclid):
The sum of the angles of a triangle is $180^\circ$.

Do you need anything more? Perhaps yes. Note that theoremhead includes the optional note to the theorem also, so that the font of the number and name of the theorem-like statement and that of the optional note are always the same. What if you need something like

Cauchy’s Theorem (Third Version). If $G$ is a simply connected open subset of $\mathbb{C}$, then for every closed rectifiable curve $\gamma$ in $G$, we have
\[
\int_\gamma f = 0.
\]

It is in such cases, that the last parameter of \newtheoremstyle is needed. Using it we can separately customize the name and number of the theorem-like statement and also the optional note. The basic syntax for setting this parameter is
{\textfamily\textbf{commands#1\textit{commands#2\textsc{commands#3}}}}
where #1 corresponds to the name of the theorem-like statement, #2 corresponds to its number and #3 corresponds to the optional note. We are here actually supplying the replacement text for a command \thmhead which has three arguments. It is as if we are defining
\renewcommand{\thmhead}[3]{...#1...#2...#3}

but without actually typing the \renewcommand{\thmhead}[3]. For example the theorem above (Cauchy’s Theorem) was produced by
\begin{eqnarray*}
\text{Cauchy's Theorem}
\end{eqnarray*}

\begin{Cauchy}[Third Version]
If $G$ is a simply connected open subset of $\mathbb{C}$, then for every closed rectifiable curve $\gamma$ in $G$, we have
$\begin{equation*}
\int_\gamma f = 0.
\end{equation*}$
\end{Cauchy}

Note that the absence of #2 in the custom-head-spec, suppresses the theorem number and that the space after #1 and the command (\mdseries #3) sets the optional note in medium size within parentheses and with a preceding space.

Now if you try to produce

Riemann Mapping Theorem. Every open simply connected proper subset of $\mathbb{C}$ is analytically homeomorphic to the open unit disk in $\mathbb{C}$.

by typing
\begin{Riemann}
Every open simply connected proper subset of $\mathbb{C}$ is analytically homeomorphic to the open unit disk in $\mathbb{C}$.
\end{Riemann}

you will get

Riemann Mapping Theorem. Every open simply connected proper subset of $\mathbb{C}$ is analytically homeomorphic to the open unit disk in $\mathbb{C}$.

Do you see what is happened? In the \texttt{\theoremstyle{diffnotenonum}}, the parameter controlling the note part of the \texttt{\theoremcaption} was defined as (\mdseries #3) and in the \texttt{\newtheorem{Riemann}}, there is no optional note, so that in the output, you get an empty “note”, enclosed in parantheses (and also with a preceding space).

To get around these difficulties, you can use the commands \texttt{\thmname}, \texttt{\thmnumber} and \texttt{\thmnote} within the \texttt{(custom-head-spec)} as

$\begin{eqnarray*}
\text{\thmname{commands#1}}$
\text{\thmnumber{commands#2}}
\text{\thmnote{commands#3}}
\end{eqnarray*}$

Each of these three commands will typeset its argument if and only if the corresponding argument in the \texttt{\thmhead} is non empty. Thus the correct way to get the Riemann Mapping theorem in Page 114 is to input
\newtheoremstyle{newnonum}{}{}\itshape{}{}\bfseries{}{}\thmnote{ (\mdseries #3)}
\theoremstyle{newnonum}
\newtheorem{newRiemann}{Riemann Mapping Theorem}
\begin{newRiemann}
Every open simply connected proper subset of $\mathbb{C}$ is analytically homeomorphic to the open unit disk in $\mathbb{C}$.
\end{newRiemann}

Then you can also produce Cauchy’s Theorem in Page 113 by typing
\theoremstyle{newnonum}
\newtheorem{newCauchy}{Cauchy’s Theorem}
\begin{newCauchy}[Third Version]
If $G$ is a simply connected open subset of $\mathbb{C}$, then for every closed rectifiable curve $\gamma$ in $G$, we have
\begin{equation*}
\int_{\gamma} f=0
\end{equation*}
\end{newCauchy}

The output will be exactly the same as that seen in Page 113. Now suppose you want to highlight certain theorems from other sources in your document, such as

\begin{cit}
[Axiom 1 in \cite{eu}]
Things that are equal to the same thing are equal to one another.
\end{cit}

This can be done as follows:
\newtheoremstyle{citing}{}{}\itshape{}{}\bfseries{}{}\thmnote{#3}
\theoremstyle{citing}
\newtheorem{cit}{}
\begin{cit}[Axiom 1 in \cite{eu}]
Things that are equal to the same thing are equal to one another.
\end{cit}

Of course, your bibliography should include the citation with label eu.

\textbf{IX.2.3. There is more!}

There are some more predefined features in \texttt{amsthm} package. In all the different examples we have seen so far, the \texttt{theorem number} comes after the \texttt{theorem name}. Some prefer to have it the other way round as in

\begin{numfirstthm}[Euclid]
The sum of the angles in a triangle is $180^\circ$.
\end{numfirstthm}

This effect is produced by the command \texttt{\swapnumbers} as shown below:
\swapnumbers
\theoremstyle{plain}
\newtheorem{numfirstthm}{Theorem}[section]
\begin{numfirstthm}[Euclid]
The sum of the angles in a triangle is $180^\circ$\end{numfirstthm}
Note that the `\swapnumbers` command is a sort of toggle-switch, so that once it is given, all subsequent theorem-like statements will have their numbers first. If you want it the other way for some other theorem, then give `\swapnumbers` again before its definition.

A quick way to suppress theoremnumbers is to use the `\newtheorem*` command as in

\begin{numlessthm}{Theorem}[section]
\swapnumbers
\end{numlessthm}

\begin{numlessthm}{Euclid}
The sum of the angles in a triangle is $180^\circ$.
\end{numlessthm}

to produce

Euclid. The sum of the angles in a triangle is $180^\circ$.

Note that this could also be done by leaving out #2 in the `custom-head-spec` parameter of `\newtheoremstyle`, as seen earlier.

We have been talking only about theorems so far, but Mathematicians do not live by theorems alone; they need proofs. The `amsthm` package contains a predefined proof environment so that the proof of a theorem-like statement can be enclosed within `\begin {proof} ... \end{proof}` commands as shown below:

\begin{thmsec}
The number of primes is infinite.
\end{thmsec}

\begin{proof}
Let $\{p_1,p_2,\dotsc p_k\}$ be a finite set of primes. Define $n=p_1 p_2 \ldots p_k+1$. Then either $n$ itself is a prime or has a prime factor. Now $n$ is either equal to or divisible by any of the primes $p_1,p_2,\dotsc p_k$ so that in either case, we get a prime different from $p_1,p_2,\dotsc p_k$. Thus no finite set of primes can include all the primes.
\end{proof}

to produce the following output

**Theorem IX.2.3.** The number of primes is infinite.

\textsc{Proof.} Let $\{p_1,p_2,\ldots p_k\}$ be a finite set of primes. Define $n = p_1 p_2 \cdots p_k + 1$. Then either $n$ itself is a prime or has a prime factor. Now $n$ is neither equal to nor is divisible by any of the primes $p_1,p_2,\ldots p_k$ so that in either case, we get a prime different from $p_1,p_2,\ldots p_k$. Thus no finite set of primes can include all the primes. \hfill \qed

There is an optional argument to the `proof` environment which can be used to change the `proofhead`. For example,

\begin{proof}[\textsc{Proof (Euclid)}]
Let $\{p_1,p_2,\ldots p_k\}$ be a finite set of primes. Define $n=p_1 p_2 \cdots p_k+1$. Then either $n$ itself is a prime or has a prime factor. Now $n$ is either equal to or divisible by any of the primes $p_1,p_2,\ldots p_k$ so that in either case, we get a prime different from $p_1,p_2,\ldots p_k$. Thus no finite set of primes can include all the primes.
\end{proof}
produces the following

**Proof (Euclid):** Let \( \{p_1, p_2, \ldots, p_k\} \) be a finite set of primes. Define \( n = p_1 p_2 \cdots p_k + 1 \). Then either \( n \) itself is a prime or has a prime factor. Now \( n \) is neither equal to nor is divisible by any of the primes \( p_1, p_2, \ldots, p_k \) so that in either case, we get a prime different from \( p_1, p_2, \ldots, p_k \). Thus no finite set of primes can include all the primes. \( \square \)

Note that the end of a proof is *automatically* marked with a \( \square \) which is defined in the package by the command \texttt{\qedsymbol}. If you wish to change it, use \texttt{\renewcommand} to redefine the \texttt{\qedsymbol}. Thus if you like the original “Halmos symbol” \( \halmos \) to mark the ends of your proofs, include

\begin{verbatim}
\newcommand{\halms}{\rule{1mm}{2.5mm}}
\renewcommand{\qedsymbol}{\halms}
\end{verbatim}

in the preamble to your document.

Again, the placement of the \texttt{\qedsymbol} at the *end* of the last line of the proof is done via the command \texttt{\qed}. The default placement may not be very pleasing in some cases as in

**Theorem IX.2.4.** The square of the sum of two numbers is equal to the sum of their squares and twice their product.

**Proof.** This follows easily from the equation

\[(x + y)^2 = x^2 + y^2 + 2xy \]

\( \square \)

It would be better if this is typeset as

**Theorem IX.2.5.** The square of the sum of two numbers is equal to the sum of their squares and twice their product.

**Proof.** This follows easily from the equation

\[(x + y)^2 = x^2 + y^2 + 2xy \]

which is achieved by the input shown below:

\begin{verbatim}
\proof
This follows easily from the equation
\begin{equation}
(x+y)^2 = x^2 + y^2 + 2xy \tag*{\qed}
\end{equation}
\renewcommand{\qed}{\halmos}
\end{proof}
\end{verbatim}

For this trick to work, you must have loaded the package *amsmath* without the \texttt{leqno} option. Or, if you prefer

**Proof.** This follows easily from the equation

\[(x + y)^2 = x^2 + y^2 + 2xy \]

Then you can use
\begin{proof}
This follows easily from the equation
\begin{equation*}
(x+y)^2 = x^2 + y^2 + 2xy \qedsymbol
\end{equation*}
\renewcommand{\qedsymbol}{}
\end{proof}

\section*{IX.3. Housekeeping}

It is better to keep all \texttt{newtheoremstyle} commands in the preamble than scattering them all over the document. Better still, you can keep them together with other customization in a personal .\texttt{sty} file and load it using the \texttt{usepackage} command in the preamble. Also, within this .\texttt{sty} file, you can divide your \texttt{newtheorem} commands into groups and preface each group with the appropriate \texttt{theoremstyle}.

\section*{Bibliography}

[1] Euclid, \textit{The Elements}, Greece 300 BC
TUTORIAL X

SEVERAL KINDS OF BOXES

The method of composing pages out of boxes lies at the very heart of \TeX{} and many \TeX{} constructs are available to take advantage of this method of composition.

A box is an object that is treated by \TeX{} as a single character. A box cannot be split and broken across lines or pages. Boxes can be moved up, down, left and right. \TeX{} has three types of boxes.

**LR** (left-right) The content of this box are typeset from left to right.

**Par** (paragraphs) This kind of box can contain several lines, which will be typeset in paragraph mode just like normal text. Paragraphs are put one on top of the other. Their widths are controlled by a user specified value.

**Rule** A thin or thick line that is often used to separate various logical elements on the output page, such as between table rows and columns and between running titles and the main text.

### X.1. LR BOXES

The usage information of four types of LR boxes are given below. The first line considers the text inside the curly braces as a box, with or without a frame drawn around it. For instance, \fbox{some words} gives some words whereas \mbox will do the same thing, but without the ruled frame around the text.

\begin{tabular}{p{0.3\textwidth}p{0.3\textwidth}p{0.3\textwidth}}
\fbox{some words} & \makebox{5cm}{some words} & \par
\framebox{5cm}{r}{some words}
\end{tabular}

In addition to the centering the text with positional argument \[c\] (the default), you can position the text flush left (\[l\]). \TeX{} also offers you an \[s\] specifier that will stretch your text from the left margin to the right margin of the box provided it contains some stretchable space. The inter-word space is also stretchable and shrinkable to a certain extent.

With \TeX{}, the above box commands with arguments for specifying the dimensions of the box allow you to make use of four special length parameters: \width, \height,
Several kinds of boxes

A few words of advice

\framebox{A few words of advice}\[6pt]
\framebox[5cm][s]{A few words of advice}\[6pt]
\framebox[1.5\textwidth]{A few words of advice}

As seen in the margin of the current line, boxes with zero width can be used to make text stick out in the margin. This effect was produced by beginning the paragraph as follows:

\makebox{0mm}{r}{$\Leftrightarrow$}

As seen in the margin of the dots

The appearance of frameboxes can be controlled by two style parameters.

\fboxrule The width of the lines comprising the box produced with the command \fbox or \framebox. The default value in all standard classes is 0.4pt.

\fboxsep The space left between the edge of the box and its contents by \fbox or \framebox. The default value in all standard classes is 3pt.

Text in a box

\fbox{Text in a box}
\setlength{\fboxrule}{2pt}\setlength{\fboxsep}{2mm}
\fbox{Text in a box}

Another interesting possibility is to raise or lower boxes. This can be achieved by the very powerful \raisebox command, which has two obligatory and two optional parameters, defined as follows:

\raisebox{\lift}{\depth}{\height}{\contents}

An example of lowered and elevated text boxes is given below.

baseline upward baseline downward baseline

baseline \raisebox{\lift}{\depth}{\height}{\contents} baseline
\raisebox{-\lift}{\depth}{\height}{\contents} baseline
As with \makebox and \framebox the \TeX{} implementation of \raisebox offers you the use of the lengths \height, \depth, \totalheight and \width in the first three arguments. Thus, to pretend that a box extends only 90% of its actual height above the baseline you could write:

\raisebox{0pt}{0.9\height}{text}

or to rotate a box around its lower left corner (instead of its reference point lying on the baseline), you could raise it by its \depth first, e.g.:

\begin{figure}
\centering
\begin{tabular}{c}
\makebox[0.5\textwidth]{\parbox{0.2\textwidth}{\fbox{Bad thing}}}
\makebox[0.5\textwidth]{\parbox{0.2\textwidth}{\fbox{Bad thing}}}
\makebox[0.5\textwidth]{\parbox{0.2\textwidth}{\fbox{Bad thing}}}
\makebox[0.5\textwidth]{\parbox{0.2\textwidth}{\fbox{Bad thing}}}
\end{tabular}
\end{figure}

\begin{verbatim}
$\textsubscript{x_1}$ \doturn{\fbox{Bad thing}}
$\textsubscript{x_2}$ \doturn{\raisebox{\depth}{\fbox{Bad thing}}}
$\textsubscript{x_3}$ \doturn{\raisebox{-\height}{\fbox{Bad thing}}}
$\textsubscript{x_4}$
\end{verbatim}

\section*{X.2. \textsc{Paragraph boxes}}

Paragraph boxes are constructed using the \parbox command or minipage environment. The text material is typeset in paragraph mode inside a box of width \texttt{width}. The vertical positioning of the box with respect to the text baseline is controlled by the one-letter optional parameter \texttt{pos} (\texttt{c}, \texttt{t}, and \texttt{b}).

The usage for \parbox command is,

\parbox{\texttt{pos}}{\texttt{width}}{\texttt{text}}

whereas that of the \texttt{minipage} environment will be:

\begin{verbatim}
\begin{minipage}{\texttt{pos}}{\texttt{width}}
...here goes the text matter...
\end{minipage}
\end{verbatim}

The center position is the default as shown by the next example. You can also observe that \TeX{} might produce wide inter-word spaces if the measure is incredibly small.

This is the contents of the left-most parbox. \hspace{\textwidth} This is the right-most parbox. Note that the typeset text looks sloppy because \TeX{} cannot nicely balance the material in these narrow columns.

The code for generating these three \parbox’s in a row is given below:

\begin{verbatim}
\parbox{.3\textwidth}{\parbox{.3\textwidth}{This is the contents of the left-most parbox.}} \hfill CURRENT LINE \hfill \parbox{.3\textwidth}{This is the right-most parbox. Note that the typeset text looks sloppy because \LaTeX{} cannot nicely balance the material in these narrow columns.}
\end{verbatim}
The minipage environment is very useful for the placement of material on the page. In effect, it is a complete mini-version of a page and can contain its own footnotes, paragraphs, and array, tabular and multicols (we will learn about these later) environments. A simple example of minipage environment at work is given below. The baseline is indicated with a small line.

\begin{minipage}{b}{.3\linewidth}
The minipage environment creates a vertical box like the parbox command. The bottom line of this minipage is aligned with the \end{minipage}\hrulefill
\begin{minipage}{c}{.3\linewidth}
middle of this narrow parbox, which in turn is \end{minipage}\hrulefill
\begin{minipage}{t}{.3\linewidth}
the top line of the right hand minipage. It is recommended that the user experiment with the positioning arguments to get used to their effects. \end{minipage}

The minipage environment creates a vertical box like the parbox command. The bottom line of this minipage is aligned with the \end{minipage}

\begin{minipage}{b}{.3\linewidth}
middle of this narrow parbox, \end{minipage}
\begin{minipage}{c}{.3\linewidth}
which in turn is \end{minipage}
\begin{minipage}{t}{.3\linewidth}
the top line of the right hand minipage. It is recommended that the user experiment with the positioning arguments to get used to their effects. \end{minipage}

X.3. PARA\sc{graph} BOXES WITH SPECIFIC HEIGHT

In \LaTeX, the syntax of the \parbox and minipage has been extended to include two more optional arguments.

\parbox\{pos\}{\textit{height}}\{\textit{inner pos}\}\{\textit{width}\}\{\textit{text}\}

is the usage for \parbox command, whereas that of the minipage environment will be:

\begin{minipage}\{pos\}{\textit{height}}\{\textit{inner pos}\}\{\textit{width}\}
... here goes the text matter ...
\end{minipage}

In both cases, \textit{height} is a length specifying the height of the box; the parameters \textit{height}, \textit{width}, \textit{depth}, and \textit{totalheight} may be employed within the \textit{emph} argument in the same way as in the \textit{width} argument of \texttt{makebox} and \texttt{framebox}.

The optional argument \textit{inner pos} states how the text is to be positioned internally, something that is only meaningful if \textit{height} has been given. Its possible values are:

\begin{tabular}{ll}
t & To push the text to the top of the box. \\
b & To shove it to the bottom. \\
c & To center it vertically. \\
s & To stretch it to fill up the whole box. \\
\end{tabular}

In the last case, we must specify the interline space we wish to have and the deviations allowed from this value as in the example below.

Note the difference between the external positioning argument \textit{pos} and the internal one \textit{inner pos}: the former states how the box is to be aligned with the surrounding text,
while the latter determines how the contents are placed within the box itself. See an example below. We frame the minipages to make it more comprehensible.

<table>
<thead>
<tr>
<th>This is a minipage with a height of 3 cm with the text aligned at the top.</th>
<th>In this minipage of same height, the text is vertically centered.</th>
<th>In this third box of same height, text is aligned at the bottom.</th>
<th>In this fourth box of same height, the text is stretched to fill in the entire vertical space.</th>
</tr>
</thead>
<tbody>
<tr>
<td>This is a minipage with a height of 3 cm with the text aligned at the top.</td>
<td>In this minipage of same height, the text is vertically centered.</td>
<td>In this third box of same height, text is aligned at the bottom.</td>
<td>In this fourth box of same height, the text is stretched to fill in the entire vertical space.</td>
</tr>
</tbody>
</table>

See the code that generated the above boxed material:

\begin{minipage}[b][3cm][t]{2cm}
This is a minipage with a height of 3 cm with the text aligned at the top.
\end{minipage}\hfill
\begin{minipage}[b][3cm][c]{2cm}
In this minipage of same height, the text is vertically centered.
\end{minipage}\hfill
\begin{minipage}[b][3cm][b]{2cm}
In this third box of same height, text is aligned at the bottom.
\end{minipage}\hfill
\begin{minipage}{b}{3cm}{s}{2cm}
\baselineskip 10pt plus 2pt minus 2pt
In this fourth box of same height, the text is stretched to fill in the entire vertical space.
\end{minipage}

In the last minipage environment the command \baselineskip gets the interline space to be 10 points text allows it to be as low as 8 points or as high as 12 points.

### X.4. Nested boxes

The box commands described above may be nested to any desired level. Including an LR box within a parbox or a minipage causes no obvious conceptual difficulties. The opposite, a parbox within an LR box, is also possible, and is easy to visualize if one keeps in mind that every box is a unit, treated by \TeX as a single character of the corresponding size.

A parbox inside an \fbox command has the effect that the entire parbox is framed. The present structure was made with

\fbox{\fbox{\parbox{.75\linewidth}{A parbox ...}}}}

This is a parbox of width .75\linewidth inside an fbox inside a second fbox, which thus produces the double framing effect.
A rule box is basically a filled-in black rectangle. The syntax for the general command is:

\texttt{\rule{\textit{lift}}{\textit{width}}{\textit{height}}}

which produces a solid rectangle of width \textit{width} and height \textit{height}, raised above the baseline by an amount \textit{lift}. Thus

\texttt{\rule{8mm}{3mm}}

generates

\begin{center}
\begin{tabular}{|c|}
\hline
\textcolor{black}{\rule{8mm}{3mm}}
\hline
\end{tabular}
\end{center}

and

\texttt{\rule{3in}{.2pt}}

generates

\begin{center}
\begin{tabular}{|c|}
\hline
\textcolor{black}{\rule{3in}{.2pt}}
\hline
\end{tabular}
\end{center}

Without an optional argument \textit{lift}, the rectangle is set on the baseline of the current line of the text. The parameters \textit{lift}, \textit{width} and \textit{height} are all lengths. If \textit{lift} has a negative value, the rectangle is set below the baseline.

It is also possible to have a rule box of zero width. This creates an invisible line with the given \textit{height}. Such a construction is called a \textit{strut} and is used to force a horizontal box to have a desired height or depth that is different from that of its contents.
XI. THE figure ENVIRONMENT

Figures are really problematical to present in a document because they never split between pages. This leads to bad page breaks which in turn leave blank space at the bottom of pages. For fine-tuning that document, the typesetter has to adjust the page breaks manually.

But \TeX{} provides floating figures which automatically move to suitable locations. So the positioning of figures is the duty of \TeX{}.

XI.1. Creating floating figures

Floating figures are created by putting commands in a figure environment. The contents of the figure environment always remains in one chunk, floating to produce good page breaks. The following commands put the graphic from figure.eps inside a floating figure:

\begin{figure}
  \centering
  \includegraphics{figure.eps}
  \caption{This is an inserted EPS graphic}
  \label{fig1}
\end{figure}

Features

- The optional \label command can be used with the \ref, and \pageref commands to reference the caption. The \label command must be placed immediately after the \caption
- If the figure environment contains no \caption commands, it produces an unnumbered floating figure.
- If the figure environment contains multiple \caption commands, it produces multiple figures which float together. This is useful in constructing side-by-side graphics or complex arrangements.
- A list of figures is generated by the \listoffigures command.
- By default, the caption text is used as the caption and also in the list of figures. The caption has an optional argument which specifies the list-of-figure entry. For example,

\caption[List Text]{Caption Text}
causes “Caption Text” to appear in the caption, but “List Text” to appear in the list of figures. This is useful when using long, descriptive captions.

- The figure environment can only be used in outer paragraph mode, preventing it from being used inside any box (such as parbox or minipage).
- Figure environments inside the paragraphs are not processed until the end of the paragraph. For example:

\begin{figure}
\end{figure}

\begin{figure}
\end{figure}

XI.1.2. Figure placement
The figure environment has an optional argument which allows users to specify possible figure locations. The optional argument can contain any combination of the letters: h, t, b, p.

- h Place the figure in the text where the figure command is located. This option cannot be executed if there is not enough room remaining on the page.
- t Place the figure at the top of the page.
- b Place the figure at the bottom of a page.
- p Place the figure on a page containing only floats.

If no optional arguments are given, the placement options default to [tbp].

When we input a float, \LaTeX will read that float and hold it until it can be placed at a better location. Unprocessed floats are those which are read by \LaTeX but have not yet been placed on the page. Though the float-placing is done by \LaTeX, sometimes the user has to invoke commands to process unprocessed floats. Following commands will do that job:
\clearpage This command places unprocessed floats and starts a new page.
\FloatBarrier This command causes all unprocessed floats to be processed. This is provided by the \texttt{placeins} package. It does not start a new page, unlike \clearpage.

Since it is often desirable to keep floats in the section in which they were issued, the section option
\usepackage[section]{placeins}
redefines the \texttt{section} command, inserting a \texttt{\FloatBarrier} command before each section. Note that this option is very strict. This option does not allow a float from the previous section to appear at the bottom of the page, since that is after the start of a new section.

The below option
\usepackage[below]{placeins}
is a less-restrictive version of the section option. It allows floats to be placed after the beginning of a new section, provided that some of the previous section appears on the page.
\afterpage/\clearpage The afterpage package provides the \texttt{\afterpage} command which executes a command at the next naturally-occurring page break.

Therefore, using \texttt{\afterpage/\clearpage} causes all unprocessed floats to be cleared at the next page break. \texttt{\afterpage/\clearpage} is especially useful when producing small floatpage figures.

\section{Customizing float placement}

The following style parameters are used by \LaTeX{} to prevent awkward-looking pages which contain too many floats or badly-placed floats.

\textbf{Float placement counters}
\topnumber The maximum number of floats allowed at the top of a text page (the default is 2).
\bottomnumber The maximum number of floats allowed at the bottom of a text page (the default is 1).
\totalnumber The maximum number of floats allowed on any one text page (the default is 3).

These counters prevent \LaTeX{} from placing too many floats on a text page. These counters do not affect float pages. Specifying a \texttt{!} in the float placement options causes \LaTeX{} to ignore these parameters. The values of these counters are set with the \texttt{\setcounter} command. For example,
\setcounter{totalnumber}{2}
prevents more than two floats from being placed on any text page.

\textbf{Figure fractions}
The commands given below control what fraction of a page can be covered by floats (where “fraction” refers to the height of the floats divided by \texttt{\textheight}). The first
three commands pertain only to text pages, while the last command pertains only to float pages. Specifying a ! in the float placement options causes \LaTeX{} to ignore the first three parameters, but \texttt{\textbackslash floatpagefraction} is always used. The value of these fractions are set by \texttt{\textbackslash renewcommand}. For example,
\begin{verbatim}
\renewcommand{\textfraction}{0.3}
\end{verbatim}

\begin{verbatim}
\textfraction \textbackslash topfraction \textbackslash bottomfraction \textbackslash floatpagefraction
\end{verbatim}

The minimum fraction of a text page which must be occupied by text. The default is 0.2, which prevents floats from covering more than 80\% of a text page.

The maximum fraction of a text page which can be occupied by floats at the top of the page. The default is 0.7, which prevents any float whose height is greater than 70\% of \texttt{\textbackslash textheight} from being placed at the top of a page.

The maximum fraction of a text page which can be occupied by floats at the bottom of the page. The default is 0.3, which prevents any float whose height is greater than 40\% of \texttt{\textbackslash textheight} from being placed at the bottom of a text page.

The minimum fraction of a float page that must be occupied by floats. Thus the fraction of blank space on a float page cannot be more than 1-\texttt{\textbackslash floatpagefraction}. The default is 0.5.

11.1.4. Using graphics in \LaTeX{}

This section shows how graphics can be handled in \LaTeX{} documents. While \LaTeX{} can import virtually any graphics format, Encapsulated PostScript (EPS) is the easiest graphics format to import into \LaTeX{}. The ‘eps’ files are inserted into the file using command \texttt{\textbackslash includegraphics}

\texttt{\includegraphics[options]{filename}}

The following options are available in \texttt{\includegraphics} command:

- \texttt{width} The width of the graphics (in any of the accepted \TeX{} units).
- \texttt{height} The height of the graphics (in any of the accepted \TeX{} units).
- \texttt{totalheight} The totalheight of the graphics (in any of the accepted \TeX{} units).
- \texttt{scale} Scale factor for the graphic. Specifying \texttt{scale = 2} makes the graphic twice as large as its natural size.
- \texttt{angle} Specifies the angle of rotation, in degrees, with a counter-clockwise (anti-clockwise) rotation being positive.

Graphics search path

By default, \LaTeX{} looks for graphics files in any directory on the \TeX{} search path. In addition to these directories, \LaTeX{} also looks in any directories specified in the \texttt{\textbackslash graphicspath} command. For example,
\begin{verbatim}
\graphicspath{\{dir1/\} dir2/}
\end{verbatim}
XI.1. The figure environment

\includegraphics[width=1in]{tex.png}

\includegraphics[height=1.5in]{tex.png}

\includegraphics[\textwidth=0.25,angle=45]{tex.png}

\includegraphics[\textwidth=0.25,angle=90]{tex.png}

tells \LaTeX to look for graphics files also in dir1/ and dir2/. For Macintosh, this becomes \graphicspath{{dir1:}{dir2:}}

Graphics extensions

The \DeclareGraphicsExtensions command tells \LaTeX which extensions to try if a file with no extension is specified in the \includegraphics command. For convenience, a default set of extensions is pre-defined depending on which graphics driver is selected. For example if dvips is used, the following graphics extensions (defined in dvips.def) are used by default

\DeclareGraphicsExtensions{.eps,.ps,.eps.gz,.ps.gz,.eps.Z}

With the above graphics extensions specified, \includegraphics{file} first looks for file.\text{eps}, then file.ps, then file.\text{eps.gz}, etc. until a file is found. This allows the graphics to be specified with

\includegraphics{file}

instead of
XI.5. Rotating and scaling objects

In addition to the \includegraphics command, the graphicx package includes four other commands which rotate and scale any \LaTeX object: text, EPS graphic, etc.

\scalebox{2}{\includegraphics{file.eps}}
\resizebox{4in}{!}{\includegraphics{file.eps}}
\rotatebox{45}{\includegraphics{file.eps}}

produces the same three graphics as
\includegraphics[scale=2]{file.eps}
\includegraphics[width=4in]{file.eps}
\includegraphics[angle=45]{file.eps}

For example, the following are produced with

\rotatebox{45}{\fbox{\LARGE{\LaTeX}}}
\rotatebox{180}{\fbox{\LARGE{\LaTeX}}}

However, the \includegraphics is preferred because it is faster and produces more efficient PostScript.

XI.2. The \texttt{table} environment

With the box elements already explained in the previous chapter, it would be possible to produce all sorts of framed and unframed tables. However, \LaTeX offers the user far more convenient ways to build such complicated structures.

XI.2.1. Constructing tables

The environments \texttt{tabular} and \texttt{tabular*} are the basic tools with which tables can be constructed. The syntax for these environments is:

\begin{tabular} \texttt{pos} \texttt{cols} \texttt{rows} \end{tabular}
\begin{tabular*} \texttt{width} \texttt{pos} \texttt{cols} \texttt{rows} \end{tabular*}

Both the above environments actually create a minipage. The meaning of the above arguments is as follows:

\texttt{pos} Vertical positioning arguments (see also the explanation of this argument for \texttt{parboxes}). It can take on the values:
1. The top line of the table is aligned with the baseline of the current external line of text.

b. The bottom line of the table is aligned with the external baseline.

With no positioning argument given, the table is centered on the external baseline.

**width** This argument applies only to the `tabular*` environment and determines its overall width. In this case, the `cols` argument must contain the @-expression (see below) `@{\extracolsep{\fill}}` somewhere after the first entry. For the other two environments, the total width is fixed by the textual content.

**cols** The column formatting argument. There must be an entry for every column, as well as possible extra entries for the left and right borders of the table or for the inter-column spacings. The possible column formatting symbols are:

| 1 | The column contents are left justified. |
| c | The column contents are centered. |
| r | The column contents are right justified. |
| {wd} | The text in this column is set into lines of width `wd` and the top line is aligned with the other columns. In fact, the text is set in a parbox with the command `\parbox[t]{wd}{column text}`. |
| *(num)*{cols} | The column format contained in `cols` is reproduced `num` times, so that `*{3}{{|c|}}` is the same as `|c|c|c|`. |

The available formatting symbols for right and left borders and for the inter-column spacing are:

| | Draws a vertical line. |
| | Draws two vertical lines next to each other. |
| | This entry is referred to as an @-expression, and inserts text in every line of the table between the two columns where it appears. |

@-expression removes the inter-column spacing that is automatically put between each pair of columns. If white space is needed between the inserted text and the next column, this must be explicitly included with `\hspace{ }` within the text of the @-expression. If the inter-column spacing between two particular columns is to be something other than the standard, this may be easily achieved by placing `@{\hspace{wd}}` between the appropriate columns in the formatting argument. This replaces the standard inter-column spacing with the width `wd`.

An `\extracolsep{wd}` within an @-expression will put extra spacing of amount `wd` between all the following columns, until countermanded by another `\extracolsep` command. In contrast to the standard spacing, this additional spacing is not removed by later @-expression. In the `tabular*` environment, there must be a command `@{\extracolsep{\fill}}` somewhere in the column format so that all the subsequent inter-column spacings can stretch out to fill the predefined table width.

If the left or right borders of the table do not consist of a vertical line, a spacing equal to half the normal inter-column spacing is added there. If this spacing is not required, it may be suppressed by including an empty @-expression `@()` at the beginning or end of the column format.
XI. Floats

Rows

Contain the actual entries in the table, each horizontal row being terminated with \\ . These rows consist of a sequence of column entries separated from each other by the & symbol. Thus each row in the table contains the same number of column entries as in the column definition cols. Some entries may be empty. The individual column entries are treated by \TeX as though they were enclosed in braces { }, so that any change in type style or size are restricted to that one column.

\hline

This command may only appear before the first row or immediately after a row termination \\ . It draws a horizontal line the full width of the table below the row that was just ended, or at the top of the table if it comes at the beginning. Two \hline commands together draw two horizontal lines with a little space between them.

\cline{m - n}

This command draws a horizontal line from the left side of column n to the right side of column m. Like \hline, it may only be given just after a row termination \\, and there may be more than one after another. The command \cline{1-3} \cline{5-7} draws two horizontal lines from column 1 to 3 and from column 5 to 7, below the row that was just ended. In each case, the full column widths are underlined.

\vline

This command draws a vertical line with the height of the row at the location where it appears. In this way, vertical lines that do not extend the whole height of the table may be inserted with a column.

\multicolumn{num}{col}{text}

This command combines the following num columns into a single column with their total width including inter-column spacing. The argument col contains exactly one of the positioning symbols l, r, c, with possible @-expressions and vertical lines ". A value of 1 may be given for num when the positioning argument is to be changed for that column in one particular row.

In this context, a ‘column’ starts with a positioning symbol l, r, or c and includes everything upto but excluding the next one. The first column also includes everything before the first positioning symbol. Thus c@{}r1" contains three columns: the first is "c@{}, the second r, and the third r".

XI.2. Table style parameters

There are a number of style parameters used in generating tables which \TeX sets to standard values. These may be altered by the user, either globally within the preamble or locally inside an environment. They should not be changed within the tabular environment.

- \tabcolsep is half the width of the spacing that is inserted between columns in the tabular and tabular* environments.
- \arrayrulewidth is the thickness of the vertical and horizontal lines within a table.
- \doublerulesep is the separation between the lines of a double rule.
- \arraystretch can be used to change the distance between the rows of a table. This is a multiplying factor, with a standard value of 1. A value of 1.5 means that the inter-row spacing is increased by 50%. A new value is set by redefining the parameter with the command:
XI.2. THE \textsc{table} ENVIRONMENT

\renewcommand{\arraystretch}{factor}

Following are the commands for changing the table style parameters that relate to dimensions:

\setlength{\tabcolsep}{\textit{dimen}}
\setlength{\arrayrulewidth}{\textit{dimen}}
\setlength{\doublerulesep}{\textit{dimen}}

XI.2.3. Example

Creating tables is much easier in practice than it would seem from the above list of formatting possibilities. This is best illustrated with an example.

The simplest table consists of rows and columns in which the text entries are either centered or justified to one side. The column widths, the spacing between the columns, and thus the entire width of the table are automatically calculated.

\begin{tabular}{|l|c|r|}
\hline
\multicolumn{3}{|c|}{Sample Tabular} \\
\hline
\textit{left} & \textit{centered} & \textit{right} \\
\textit{aligned} & \textit{items} & \textit{aligned} \\
\textit{items} & \textit{items} & \textit{items} \\
\textit{left items} & \textit{centered} & \textit{right aligned} \\
\hline
\end{tabular}

See the code that generated the table above.

\begin{tabular}{|l|c|r|}
\hline
\multicolumn{3}{|c|}{Sample Tabular} \\
\hline
col head & col head & col head \\
\hline
\textit{left} & \textit{centered} & \textit{right} \\
\textit{aligned} & \textit{items} & \textit{aligned} \\
\textit{items} & \textit{items} & \textit{items} \\
\textit{left items} & \textit{centered} & \textit{right aligned} \\
\hline
\end{tabular}

The discussion on tables doesn’t conclude with this chapter, instead more bells and whistles are to be discussed, such as long tables (tables that span multiple pages), how to repeat the column headings and special footlines in all multipaged tables, color tables and also a few other embellishments, which the scientific community at large might require in their document preparation.

XI.2.4. Exercise

Here is an exercise you can try.
Plan for \TeX{} Users Group 2001–2003

<table>
<thead>
<tr>
<th>Project</th>
<th>No.</th>
<th>Name</th>
<th>Year</th>
<th>2001</th>
<th>2002</th>
<th>2003</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Journal costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>\TeX{}Live production costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signature</th>
<th>Authorization</th>
</tr>
</thead>
</table>
CROSS REFERENCES IN $\LaTeX$

XII.1. WHY CROSS REFERENCES?

Cross reference is the technical term for quoting yourself. This is what you do when you say something like, “As I said earlier,...”. More seriously, in a written article you may often have occasion to refer the reader to something mentioned earlier (or sometimes to something yet to be said) in the same document. Thus you may have explained a new term in the second section of your article and when you use this term again in the fourth section, it is a matter of courtesy to the reader to point to the explanation. Again, in a mathematics article, you may have to cite an earlier result in the proof of the current result.

Such cross referencing can be done by hand, but if you revise your document and insert some new sections (or theorems) then changing all cross references manually is no easy task. It is always better to automate such tedious tasks. (After all what’s a computer for, if not to do such mundane jobs?)

XII.2. LET $\LaTeX$ DO IT

The basic method of using cross references (see Section XII.1 for what we mean by cross reference) in $\LaTeX$ is quite simple. Suppose that somewhere in the second section of your article, you want to refer to the first section. You assign a key to the first section using the command

\begin{verbatim}
\section{section name}\label{key}
\end{verbatim}

and at the point in the second section where the reference is to be made, you type the command

\begin{verbatim}
\ref{key}
\end{verbatim}

Thus the reference “see Section XII.1...” in the first sentence of this section was produced by including the command \label{intro} in the command for the first section as

\begin{verbatim}
\section{Why cross references}\label{intro}
\end{verbatim}

and the command \ref{intro} at the place of reference in the second section as

... (see Section \ref{intro} for...)

Okay, the example is a bit silly, since the actual reference here is not really necessary, but you get the general idea, don’t you? Incidentally, the \label{key} for a section need not be given immediately after the \section{section name}. It can be given anywhere within the section.
The first time you run \LaTeX{} on a file named, say, \texttt{myfile.tex} containing cross references, the reference information is written in an auxiliary file named \texttt{myfile.aux} and at the end of the run \LaTeX{} prints a warning

\textit{LaTeX Warning: There were undefined references.}

\textit{LaTeX Warning: Label(s) may have changed.  Rerun to get cross-references right.}

A second run gets the references right. The same thing happens when you’ve changed the reference information in any way, say, by adding a new section.

Though the \texttt{key} in \texttt{\label{key}} can be any sequence of letters, digits or punctuation characters, it is convenient to use some mnemonic (such as \texttt{\label{limcon}} for a section entitled “Limits and Continuity” rather than \texttt{\label{sec@#*?!}}). Also, when you make a reference, it’s better to type \texttt{\ref{limcon}} (notice the tie?) than \texttt{\ref{limcon}} to prevent the possibility of the reference number falling off the edge as in “…see Section X.II for further details….”.

In addition to sectioning commands such as \texttt{\chapter} or \texttt{\section}, reference can also be made to an \texttt{\item} entry in an \texttt{enumerate} environment, by attaching a \texttt{\label}. For example the input

\begin{verbatim}
In the classical \textbf{\emph{syllogism}}
\begin{enumerate}
\item All men are mortal. \label{pre1}
\item Socrates is a man. \label{pre2}
\item So Socrates is a mortal. \label{con}
\end{enumerate}
\end{verbatim}

\textit{In the classical syllogism}

(1) All men are mortal.
(2) Socrates is a man.
(3) So Socrates is a mortal.

Statements (1) and (2) are the \emph{premises} and statement (3) is the conclusion.

You must be a bit careful about references to tables or figures (technically, “floats”). For them, the \texttt{\label} command should be given after the \texttt{\caption} command or in its argument, as in the example below:

\begin{verbatim}
\begin{table}[h]
\begin{center}
\setlength{\extrarowheight}{5pt}
\begin{tabular}{|c|c|c|c|}
\hline
Value of $x$ & 1 & 2 & 3 \\
\hline
Value of $y$ & 1 & 8 & 27 \\
\hline
\end{tabular}
\caption{Observed values of $x$ and $y$} \label{tabxy}
\end{center}
\end{table}
\end{verbatim}

\textit{In the classical syllogism}

(1) All men are mortal.
(2) Socrates is a man.
(3) So Socrates is a mortal.

Statements (1) and (2) are the \emph{premises} and statement (3) is the conclusion.

You must be a bit careful about references to tables or figures (technically, “floats”). For them, the \texttt{\label} command should be given after the \texttt{\caption} command or in its argument, as in the example below:

\begin{verbatim}
\begin{table}[h]
\begin{center}
\setlength{\extrarowheight}{5pt}
\begin{tabular}{|c|c|c|}
\hline
Value of $x$ & 1 & 2 & 3 \\
\hline
Value of $y$ & 1 & 8 & 27 \\
\hline
\end{tabular}
\caption{Observed values of $x$ and $y$} \label{tabxy}
\end{center}
\end{table}
\end{verbatim}

\textit{In the classical syllogism}

(1) All men are mortal.
(2) Socrates is a man.
(3) So Socrates is a mortal.

Statements (1) and (2) are the \emph{premises} and statement (3) is the conclusion.
Two possible relations between $x$ and $y$ satisfying the data in Table\ref{tab:xy} are $y=x^3$ and $y=6x^2-11x+6$.

This produces the following output:

<table>
<thead>
<tr>
<th>Value of $x$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value of $y$</td>
<td>1</td>
<td>8</td>
<td>27</td>
</tr>
</tbody>
</table>

Table XII.1: Observed values of $x$ and $y$

Two possible relations between $x$ and $y$ satisfying the data in Table XII.1 are $y = x^3$ and $y = 6x^2 - 11x + 6$.

You can think of a \caption command within a figure or table environment as a sort of sectioning command within the environment. Thus you can have several \caption and \label pairs within a single figure or table environment.

You can also make forward references in exactly the same way by \ref-ing to the key of some succeeding \label such as "see Subsection XII.2.1 for a discussion of cross references in mathematics."

XII.2.1. Cross references in math

Mathematical documents abound in cross references. There are references to theorems and equations and figures and whatnot. The method of reference is exactly as before. Thus if you’ve defined \newtheorem{theorem}[subsection], then after typing

\begin{theorem}\label{diffcon}
Every differentiable function is continuous
\end{theorem}

you get

XII.2.1.1 Theorem. Every differentiable function is continuous

and you can type elsewhere in the document

The converse of Theorem\ref{diffcon} is false.

to get

The converse of Theorem XII.2.1.1 is false.

References can be made to equations as in the following examples:

\begin{equation}\label{sumsq}
(x+y)^2=x^2+2xy+y^2
\end{equation}

Changing $y$ to $-y$ in Equation\ref{sumsq} gives the following
If you load the package `amsmath`, you can use the command `\eqref` instead of `\ref` to make a reference to an equation. This automatically supplies the parentheses around the equation number and provides an italic correction before the closing parenthesis, if necessary. For example,

Equation \eqref{sumsq} gives the following ..........

produces

Equation XII.1 gives the following ............

References can be made to individual equations in multiline displays of equations produced by such environments as `align` or `gather` (defined in the `amsmath` package). The `\label` command can be used within such a structure for subnumbering as in the example below:

\begin{align}
(x+y)^2 &= x^2 + 2xy + y^2 \label{sum} \\
(x-y)^2 &= x^2 - 2xy + y^2 \\tag{\ref{sum}a}
\end{align}

XII.3. Pointing to a Page—The Package `varioref`

In making a reference to a table or an equation, it is more to convenient (for the reader, that is) to give the page number of the reference also. The command

\`\pageref{key}\`

typesets the number of the page where the command `\label(key)` was given. Thus for example

see Table `\ref{tabxy}` in page \`\pageref{tabxy}\`

in this document produces

see Table XII.1 in page 137

To avoid the tedium of repeated by typing

\`\ref{key}\` on page \`\pageref{key}\`

you can define the macro
and use \fullref for such references. But the trouble is that at times the referred object and the reference to it fall on the same page (with \LaTeX you never know this till the end) so that you get a reference to the page number of the very page you are reading, which looks funny. This can be avoided by using the package varioref. If you load this package by including \usepackage{varioref} in your preamble, then you can use the command

\vref{\label{key}}

to refer to an object you’ve marked with \label{key} elsewhere in the document. The action of \vref varies according to the page(s) where the referred object and the references are typeset by \LaTeX in the final output.

(1) If the object and the reference are on the same page, \vref produces only a \ref suppressing \pageref so that only the number pointing to the object is typeset, without any reference to the page number.

(2) If the object and the reference are on different pages whose numbers differ by more than one, \vref produces both \ref and \pageref.

(3) If the object and the reference fall on pages whose numbers differ by one (that is, on successive pages), \vref produces \ref followed by the phrase “on the preceding page” or “on the following page” depending on whether the object or the reference occurs first. Moreover, in the next occurrence of \vref in a situation of the same type, the phrases are changed to “on the next page” and the “page before” respectively.

This is the default behavior of \vref in the \article documentclass. If the \article class is used with the twoside option or if the documentclass book is used, then the behavior in Case (3) above is a bit different.

(1) If the object and the reference fall on the two sides of the same leaf, the behavior of \vref is as in (3) above.

(2) If the object and the reference fall on pages forming a double spread (that is, a page of even number followed by the next page), then \vref produces \ref followed by the phrase “on the facing page”. Moreover, in the next occurrence of \vref in a situation of the same type, the phrases are changed to “on the preceding page” and “on the next page” respectively.

The phrases used in the various cases considered above can be customized by redefining the commands used in generating them. For the \article class without the twoside option, reference to the previous page uses the command \reftextbefore and reference to the next page uses \reftextafter. In the case of the \article class with the twoside option or the book class, the commands \reftextfaceafter and \reftextfacebefore are used in the case of reference to a page in a double spread. The default definitions of these commands are given below. In all these, the two arguments of the command \reftextvario are phrases alternatively used in the repeated use of the reference as mentioned above.

\newcommand{\reftextbefore}{on the \reftextvario{preceding page}{page before}}
\newcommand{\reftextafter}{on the \reftextvario{following}{next} page}
\newcommand{\reftextfacebefore}{on the \reftextvario{facing}{preceding} page}
\newcommand{\reftextfaceafter}{on the \reftextvario{facing}{next}{page}}
You can customize the phrases generated in various situations by redefining these with phrases of your choice in the arguments of \reftextvario.

If you want to refer only to a page number using \varioref, you can use the command

\vpageref{key}

to produce the page number of the object marked with \label{key}. The phrases used in the various special cases are the same as described above, except that when the referred object and the reference fall on the same page, either the phrase “on this page” or “on the current page” is produced. The command used to generate these is \reftextcurrent whose default definition is

\newcommand{\reftextcurrent}{on \reftextvario{this}{the current} page}

You can change the phrases “this” and “the current” globally by redefining this command. You can also make some local changes by using the two optional arguments that \vpageref allows. Thus you can use the command

\vpageref[same page phrase][other page phrase]{key}

to refer to the page number of the object marked with \label{key}. The same page phrase will be used if the object and the reference fall on the same page and the phrase other page phrase will be used, if they fall on different pages. Thus for example, the command

see the \vpageref[above table][table]{tabxy}

given in this document will produce

<table>
<thead>
<tr>
<th>see the above table</th>
</tr>
</thead>
</table>

if the reference occurs on the same page as Table XII.1 and

<table>
<thead>
<tr>
<th>see the table on page 137</th>
</tr>
</thead>
</table>

if they fall on different pages.

XII.4. POINTING OUTSIDE—THE PACKAGE XR

Sometimes you may want to refer to something in a document other than the one you are working on. (This happens, for instance if you keep an article as separate files.) The package xr allows such external references.

If you want to refer to objects in a file named other.tex in your current document, load the package xr and set the external document as other.tex using the commands

\usepackage{xr} \externaldocument{other}

in the preamble of the current document. Then you can use the \ref and \pageref to refer to anything that has been marked with the \label command in either the current document or other.tex. Any number of such external documents can be specified.

If the same key is used to mark different objects in two such documents, there’ll be a conflict. To get over this, you can use the optional argument available in \externaldocument command. If you say

\externaldocument[a-]{other}

then a reference to \label{key} in other.tex could be made by \ref{a-key}. The prefix need not be a-; it can be any convenient string.
One of the conveniences of using keys for cross references is that you need not keep track of the actual numbers, but then you’ll have to remember the keys. You can produce the list of keys used in a document by running \TeX on the file lablst.tex. In our system, we do this by first typing

\texttt{latex lablst}

\TeX responds as follows:

\begin{verbatim}
***************
* Enter input file name
* without the .tex extension:
***************
\lablstfile=

We type in the file name as \texttt{cref} which is the source of this document and is presented with another query.

***************
* Enter document class used in file cref.tex
* with no options or extension:
***************
\lablstclass=

So we type \texttt{article}. And is asked

***************
* Enter packages used in file cref.tex
* with no options or extensions:
***************
\lablstpackages=

Here only those packages used in the article which define commands used in section titles etc. need be given. So we type

\texttt{amsmath, array, enumerate}

This produces a file \texttt{lablst.dvi} which can be viewed to see a list of keys used in the document.

Finally if your text editor is GNU Emacs, then you can use its RefTeX package to automate generation, insertion and location of keys at the editing stage.
Footnotes, Marginpars, and Endnotes

\LaTeX{} has facilities to typeset “inserted” text, such as footnotes, marginal notes, figures and tables. This chapter looks more closely at different kinds of notes.

\section{Footnotes}

Footnotes are generated with the command

\footnote{footnote_text}

which comes immediately after the word requiring an explanation in a footnote. The text footnote_text appears as a footnote in a smaller typeface at the bottom of the page. The first line of the footnote is indented and is given the same footnote marker as that inserted in the main text. The first footnote on a page is separated from the rest of the page text by means of a short horizontal line.

The standard footnote marker is a small, raised number\(^1\), which is sequentially numbered.

Footnotes produced with the \footnote command inside a minipage environment use the mpfootnote counter and are typeset at the bottom of the parbox produced by the minipage.\(^2\).

However, if you use the \footnotemark command in a minipage it will produce a footnote mark in the same style and sequence as the main text footnotes—i.e., stepping the mpfootnote counter and using the \thefootnote command for the representation. This behavior allows you to produce a footnote inside your minipage that is typeset in sequence with the main text footnotes at the bottom of the page: you place a \footnotemark inside the minipage and the corresponding \footnotetext after it. See below:

\begin{minipage}{5cm}
Footnotes in a minipage are numbered using lowercase letters.\(^3\)
This text references a footnote at the bottom of the page.\(^4\)
\end{minipage}
\footnotetext{At bottom of page}

The footnote numbering is incremented throughout the document for the article class, where it is reset to 1 for each new chapter in the report and book classes.

\(^1\)See how the footnote is produced: “... raised number \footnote{See how the footnote is produced: ... \}.\(^2\)With nested minipages, the footnote comes after the next \end{minipage} command, which could be at the wrong place.\(^3\)At bottom of page.
XIII.1.1. Footnotes in tabular material

Footnotes appearing inside tabular material are not typeset by standard \TeX. Only \texttt{tabularx} and \texttt{longtable} environments will treat footnotes correctly. But footnotes used in these tables won’t appear just following the tables, but would appear at the bottom of the page just like the footnotes used in the text. But in \texttt{longtable} you can place the footnotes as table notes by placing the \texttt{longtable} in a \texttt{minipage}. See below:

\begin{minipage}{.47\textwidth}
\renewcommand{\thefootnote}{\thempfootnote}
\begin{longtable}{ll}
\caption{PostScript type 1 fonts} \\
Courier\footnote{Donated by IBM.} & cour, courb, courbi, couri \\
Nimbus\footnote{Donated by URW GmbH.} & ummr, unmrs \\
URW Antiqua\footnote{Donated by URW GmbH.} & uaqrrc \\
URW Grotesk\footnote{Donated by URW GmbH.} & ugqp \\
Utopia\footnote{Donated by Adobe.} & putb, putbi, putr, putri
\end{longtable}
\end{minipage}

You can also put your \texttt{tabular} or \texttt{array} environment inside a \texttt{minipage} environment, since in that case footnotes are typeset just following that environment. Note the redefinition of \texttt{\thefootnote} that allows us to make use of the \texttt{\footnotemark} command inside the \texttt{minipage} environment. Without this redefinition \texttt{\footnotemark} would have generated a footnote mark in the style of the footnotes for the main page.

\begin{minipage}{.5\linewidth}
\renewcommand{\thefootnote}{\thempfootnote}
\begin{tabular}{ll}
\multicolumn{2}{c}{\textbf{PostScript type 1 fonts}} \\
Courier\footnote{Donated by IBM.} & cour, courb, courbi, couri \\
Charter\footnote{Donated by Bitstream.} & bchb, bchbi, bchr, bchri \\
Nimbus\footnote{Donated by URW GmbH.} & ummr, unmrs \\
URW Antiqua\footnotemark\[\value{mpfootnote}\] & uaqrrc \\
URW Grotesk\footnotemark\[\value{mpfootnote}\] & ugqp \\
Utopia\footnote{Donated by Adobe.} & putb, putbi, putr, putri
\end{tabular}
\end{minipage}
XIII.1. Footnotes

PostScript type 1 fonts

<table>
<thead>
<tr>
<th>Font</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Courier</td>
<td>cour, courb, courbi, couri</td>
</tr>
<tr>
<td>Charter</td>
<td>bchb, bchbi, bchr, bchri</td>
</tr>
<tr>
<td>Nimbus</td>
<td>unmr, unmrs</td>
</tr>
<tr>
<td>URW Antiqua</td>
<td>uaqrc</td>
</tr>
<tr>
<td>URW Grotesk</td>
<td>ugqp</td>
</tr>
<tr>
<td>Utopia</td>
<td>putb, putbi, putr, putri</td>
</tr>
</tbody>
</table>

\[\text{\textbullet}^a\text{ Donated by IBM.}\]
\[\text{\textbullet}^b\text{ Donated by Bitstream.}\]
\[\text{\textbullet}^c\text{ Donated by URW GmbH.}\]
\[\text{\textbullet}^d\text{ Donated by Adobe.}\]

Of course this approach does not automatically limit the width of the footnotes to the width of the table, so a little iteration with the \texttt{minipage} width argument might be necessary.

Another way to typeset table notes is with the package \texttt{threeparttable} by Donald Arseneau. This package has the advantage that it indicates unambiguously that you are dealing with notes inside tables and, moreover, it gives you full control of the actual reference marks and offers the possibility of having a caption for our tabular material. In this sense, the \texttt{threeparttable} environment is similar to the nonfloating \texttt{table} environment.

\begin{threeparttable}
\caption{	extbf{PostScript type 1 fonts}}
\begin{tabular}{ll}
Yang\textbullet{a} & cour, courb, courbi, couri \\
Charter\textbullet{b} & bchb, bchbi, bchr, bchri \\
Nimbus\textbullet{c} & unmr, unmrs \\
URW Antiqua\textbullet{c} & uaqrc \\
URW Grotesk\textbullet{c} & ugqp \\
Utopia\textbullet{d} & putb, putbi, putr, putri \\
\end{tabular}
\begin{tablenotes}
\item[a] Donated by IBM.
\item[b] Donated by Bitstream.
\item[c] Donated by URW GmbH.
\item[d] Donated by Adobe.
\end{tablenotes}
\end{threeparttable}

Table 14.2: PostScript type 1 fonts

<table>
<thead>
<tr>
<th>Font</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Courier</td>
<td>cour, courb, courbi, couri</td>
</tr>
<tr>
<td>Charter</td>
<td>bchb, bchbi, bchr, bchri</td>
</tr>
<tr>
<td>Nimbus</td>
<td>unmr, unmrs</td>
</tr>
<tr>
<td>URW Antiqua</td>
<td>uaqrc</td>
</tr>
<tr>
<td>URW Grotesk</td>
<td>ugqp</td>
</tr>
<tr>
<td>Utopia</td>
<td>putb, putbi, putr, putri</td>
</tr>
</tbody>
</table>

\[\text{\textbullet}^a\text{ Donated by IBM.}\]
\[\text{\textbullet}^b\text{ Donated by Bitstream.}\]
\[\text{\textbullet}^c\text{ Donated by URW GmbH.}\]
\[\text{\textbullet}^d\text{ Donated by Adobe.}\]
XIII.1.2. Customizing footnotes

If the user wishes the footnote numbering to be reset to 1 for each \section command with the article class, this may be achieved by putting

\setcounter{footnote}{0}

before every section or using the following command at preamble:

\@addtoreset{footnote}{section}

The internal footnote counter has the name footnote. Each call to \footnote increments this counter by one and prints the new value in Arabic numbering as the footnote marker.

A different style of marker can be implemented with the command

\renewcommand{\thefootnote}{{\fnsymbol{footnote}}}

where \fnsymbol is one of nine symbols:

\[ \star \dagger \ddagger \S \P \| \ast \dagger\dagger \ddagger\ddagger \]

It is up to the user to see that the footnote counter is reset to zero sometime before the tenth \footnote call is made. If the user wants to add values above nine, then he has to edit the definition of \fnsymbol. See an example, which allows up to 12 footnotes without resetting the counter:

\makeatletter
\def\@fnsymbol#1{\ensuremath{\ifcase#1\or \star\or \dagger\or \ddagger\or \mathsection\or \mathparagraph\or \|\or \ast\or \dagger\dagger\or \ddagger\ddagger\or \mathsection\mathsection\or \mathparagraph\mathparagraph\or \|\|\else@ctrerr\fi}}
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
\makeatother

An optional argument may be added to the \footnote command:

\footnote[num]{footnote_text}

where \texttt{num} is a positive integer that is used instead of the value of the footnote counter for the marker. In this case, the footnote counter is not incremented. For example*

\makeatletter
\def\@fnsymbol#1{\ensuremath{\ifcase#1\or \star\or \dagger\or \ddagger\or \mathsection\or \mathparagraph\or \|\or \ast\or \dagger\dagger\or \ddagger\ddagger\or \mathsection\mathsection\or \mathparagraph\mathparagraph\or \|\|\else@ctrerr\fi}}
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
\makeatother

\footnote[7]{The 7\textsuperscript{th} symbol appears as the footnote marker.}

\renewcommand{\thefootnote}{\arabic{footnote}}

where the last line is necessary to restore the footnote marker style to its standard form. Otherwise, all future footnotes would be marked with symbols and not with numbers.

XIII.1.3. Footnote style parameters

The appearance of the standard footnote can be changed by customizing the parameters listed below:

- \footnotesize The font size used inside footnotes.

\footnotemark[1]This command will only work within \makeatletter and \makeatother.

\footnotemark[2]The 7\textsuperscript{th} symbol appears as the footnote marker.
The height of a strut placed at the beginning of every footnote. If it is greater than the `\baselineskip` used for `\footnotesize`, then additional vertical space will be inserted above each footnote.

`\skip\footins` A low-level TeX command that defines the space between the main text and the start of the footnotes. You can change its value with the `\setlength` or `\addtolength` commands by putting `\skip\footins` into the first argument, e.g.,

`\addtolength{\skip\footins}{3mm}`

`\footnoterule` A macro to draw the rule separating footnotes from the main text. It is executed right after the vertical space of `\skip\footins`. It should take zero vertical space, i.e., it should use a negative skip to compensate for any positive space it occupies, for example:

`\renewcommand{\footnoterule}{\vspace*{-3pt}\rule{.4\columnwidth}{0.4pt}\vspace*{2.6pt}}`

You can also construct a fancier “rule” e.g., one consisting of a series of dots:

`\renewcommand{\footnoterule}{\vspace*{-3pt}\qquad\dotfill\qquad\vspace*{2.6pt}}`

**xiii.2. Marginal notes**

`\marginpar{left-text}{right-text}`

The `\marginpar` command generates a marginal note. This command typesets the text given as an argument in the margin, the first line at the same height as the line in the main text where the `\marginpar` command occurs. The marginal note appearing here was generated with

`\marginpar{This is a marginal note}. The ...`

When only the mandatory argument `right-text` is specified, then the text goes to the right margin for one-sided printing; to the outside margin for two-sided printing; and to the nearest margin for two-column formatting. When you specify an optional argument, it is used for the left margin, while the second (mandatory) argument is used for the right.

There are a few important things to understand when using marginal notes. First, `\marginpar` command does not start a paragraph, that is, if it is used before the first word of a paragraph, the vertical alignment may not match the beginning of the paragraph. Secondly, if the margin is narrow, and the words are long (as in German), you may have to precede the first word by a `\hspace{0pt}` command to allow hyphenation of the first word. These two potential problems can be eased by defining a command `\marginlabel{text}`, which starts with an empty box `\mbox{}`, typesets a marginal note ragged left, and adds a `\hspace{0pt}` in front of the argument.

`\newcommand{\marginlabel}[1]{\mbox{}\marginpar{\raggedleft\hspace{0pt}#1}}`

By default, in one-sided printing the marginal notes go on the outside margin. These defaults can be changed by the following declarations:

`\reversemarginpar` Marginal notes go into the opposite margin with respect to the default one.

`\normalmarginpar` Marginal notes go into the default margin.
XIII.2.1. Uses of marginal notes

\marginpar{} can be used to draw attention to certain text passages by marking them with a vertical bar in the margin. The example marking this paragraph was made by including

\marginpar{\rule[-10.5mm]{1mm}{10mm}}

in the first line. By defining a macro \query as shown below

\def\query#1#2{\underline{#1}\marginpar{#2}}

we can produce queries. For example \TeX. This query is produced with the following command.

For example \query{\LaTeX}{Hey! \ Look}{}. This ...

XIII.2.2. Style parameters for marginal notes

The following style parameters may be changed to redefine how marginal notes appear:

\marginparwidth Determines the width of the margin box.
\marginparsep Sets the separation between the margin box and the edge of the main text.
\marginparpush Is the smallest vertical distance between two marginal notes.

These parameters are all lengths and are assigned new values as usual with the \setlength command.

XIII.3. ENDNOTES

Scholarly works usually group notes at the end of each chapter or at the end of the document. These are called endnotes. Endnotes are not supported in standard \TeX, but they can be created in several ways.

The package endnotes (by John Lavagnino) typesets endnotes in a way similar to footnotes. It uses an extra external file, with extension .ent, to hold the text of the endnotes. This file can be deleted after the run since a new version is generated each time.

With this package you can output your footnotes as endnotes by simply giving the command:

\renewcommand{\footnote}{\endnote}

The user interface for endnotes is very similar to the one for footnotes after substituting the word “foot” for “end”. The following example shows the principle of the use of endnotes, where you save text in memory with the \endnote command, and then typeset all accumulated text material at a point in the document controlled by the user.

This is simple text.\footnote{The first endnote.} This is simple text.\footnote{The second endnote.} This is simple text.\footnote{The third endnote.}

This is some more simple text

This is simple text.\endnote{The first endnote.} This is simple text.\endnote{The second endnote.} This is simple text.\endnote{The third endnote.}

This is some more simple text

\theendnotes\bigskip

This is some more simple text
GNU FREE DOCUMENTATION LICENSE

Version 1.2, November 2002

Copyright © 2000, 2001, 2002 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document free in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

(1) APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.
A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, TeXinfo input format, \TeX\ input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it
remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

(2) VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

(3) COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well
before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

(4) MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

(A) Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

(B) List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

(C) State on the Title page the name of the publisher of the Modified Version, as the publisher.

(D) Preserve all the copyright notices of the Document.

(E) Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

(F) Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

(G) Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

(H) Include an unaltered copy of this License.

(I) Preserve the section Entitled “History”, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled “History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

(J) Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

(K) For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

(L) Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

(M) Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.

(N) Do not retile any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
(O) Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

(5) COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements.”

(6) COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various
documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

(7) AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

(8) TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

(9) TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (C) year your name. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with... Texts.” line with this:

with the Invariant Sections being list their titles, with the Front-Cover Texts being list, and with the Back-Cover Texts being list.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.