Indian TEX Users Group

URL: http://www.river-valley.com/tug

On-line Tutorial on LATEX

The Tutorial Team
Indian TEX Users Group, SJP Buildings, Cotton Hills
Trivandrum 695014, INDIA
2000

Prof. (Dr.) K. S. S. Nambooripad, *Director, Center for Mathematical Sciences, Trivandrum, (Editor)*;
Dr. E. Krishnan, *Reader in Mathematics, University College, Trivandrum*; T. Rishi, *Focal Image (India) Pvt. Ltd., Trivandrum*; L. A. Ajith, *Focal Image (India) Pvt. Ltd., Trivandrum*; A. M. Shan, *Focal Image (India) Pvt. Ltd., Trivandrum*; C. V. Radhakrishnan, *River Valley Technologies, Software Technology Park, Trivandrum* constitute the TUG*India* Tutorial team

This document is generated from LATEX sources compiled with pdfLATEX v. 14e in an INTEL Pentium III 700 MHz system running Linux kernel version 2.2.14-12. The packages used are hyperref.sty and pdfscreen.sty

Gentle Reader,

Generation of letter forms by mathematical means was first tried in the fifteenth century; it became popular in the sixteenth and seventeenth centuries; and it was abandoned (for good reasons) during the eighteenth century. Perhaps the twentieth century will turn out to be the right time for this idea to make a comeback, now that mathematics has advanced and computers are able to do the calculations.

Modern printing equipment based on raster lines—in which metal "type" has been replaced by purely combinatorial patterns of zeroes and ones that specify the desired position of ink in a discrete way—makes mathematics and computer science increasingly relevant to printing. We now have the ability to give a completely precise definition of letter shapes that will produce essentially equivalent results on all raster-based machines. Moreover, the shapes can be defined in terms of variable parameters; computers can "draw" new fonts of characters in seconds, making it possible for designers to perform valuable experiments that were previously unthinkable.

— *Donald Erwin Knuth,* The METAFONT book

Contents

1 Introduction 3

1.1 The Concept of Generic Markup 3

1.2 A Short History of TEX 3

1.3 What is LATEX then? 5

1.4 Getting Started 5

1.4.1 A typical LATEX input file 6

1 Introduction

1.1 The Concept of Generic Markup

Originally markup was the annotation of manuscripts of a copy editor telling the typesetter how to format the manuscript. It consisted of handwritten notes such as '*set this heading in 12 point Helvetica italic on a 10 point body, justified on a 22 pica slug with indents of 1 em on the left and none on the right*.' With the advent of computers, these marks could be coded electronically using a special coding system and people started inventing their own markup schemes. The following low level formatting commands used to instruct a computer for *carriage return*, *center the following text*, and *go to the next page* are a typical example:

```
.pa ; .sp 2 ; .ce ; .bd
Title of the chapter
.sp
```

In another markup scheme it will be like as given hereunder:

```
\vfill\eject\begingroup\bf\obeylines\vskip 20pt
\hfil Title of the chapter
\vskip 10pt\endgroup\bigskip
```

Documents created with such specific markup became difficult for typesetting systems to cope up. A movement was started to create a standard markup language, which all typesetting vendors would be persuaded to accept as input. Thus came the Generic Markup Language (GML) which later on developed into Standard Generalized Markup Language (SGML) and now a subset of which known as Extensible Markup Language (XML) is poised to take up the World Wide Web.

However, the development of SGML moved towards representing the documents in an exchangeable format aiming at 'publishing in its broadest sense, ranging from single medium conventional publishing to multimedia data base publishing'. SGML can also be used in office document processing when the benefits of human readability and interchange with publishing systems as required. It is a meta-language for defining infinite variety of markup languages and is not concerned with the formatting of marked-up documents., *i.e.*, there is no layout tags.

This void is filled by the advent of TEX which combines the balance of generic markup and layout specific support. The class file mechanism followed in LATEX makes it possible to produce the same source document in different layouts, while enough bells and whistles are available to fine-tune important documents for producing the highest quality.

1.2 A Short History of TEX

Donald E. Knuth

TEX (= tau epsilon chi, and pronounced similar to 'tech') is a computer language designed by Donald Erwin Knuth of Stanford University, for use in typesetting; in particular, for typesetting math and other technical (from Greek 'techne' = art/craft, the stem of 'technology') material.

In the late 1970s, Donald Knuth was revising the second volume of his multi-volume opus, *The Art of Computer Programming*, got the galleys, looked at them, and said (approximately) 'bleccch'! he had just received his first samples of the new computer typesetting, and its quality was so far below that of the first edition of Volume 2 that he couldn't stand it. He thought for awhile, and said (approximately), "I'm a computer scientist; I ought to be able to do something about this", so he set out to learn what were the traditional rules for typesetting math, what constituted good typography, and (because the fonts of symbols that he needed really didn't exist) as much as he could about type design. He figured this would take about 6 months. (Ultimately, it took nearly 10 years, but along the way he had lots of help from some people who should be well known to readers of this list – Hermann Zapf, Chuck Bigelow, Kris Holmes, Matthew Carter and Richard Southall are acknowledged in the introduction to Volume E, *Computer Modern Typefaces*, of the Addison-Wesley *Computers & Typesetting* book series.)

A year or so after he started, Knuth was invited by the American Mathematical Society (AMS) to present one of the principal invited lectures at their annual meeting. This honor is awarded to significant academic researchers who (mostly) were trained as mathematicians, but who have done most of their work in not strictly mathematical areas (there are a number of physicists, astronomers, etc., in the annals of this lecture series as well as computer scientists); the lecturer can speak on any topic (s)he wishes, and Knuth decided to speak on computer science in the service of mathematics. The topic he presented was his new work on TEX (for typesetting) and METAFONT (for developing fonts for use with TEX). He presented not only the roots of the typographical concepts, but also the mathematical notions (e.g., the use of bezier splines to shape glyphs) on which these two programs are based. The programs sounded like they were just about ready to use, and quite a few mathematicians, including the chair of the Mathematical Society's board of trustees, decided to take a closer look. As it turned out, TEX was still a lot closer to a research project than to an industrial strength product, but there were certain attractive features:

- it was intended to be used directly by authors (and their secretaries) who are the ones who really know what they are writing about;
- it came from an academic source, and was intended to be available for no monetary fee (nobody said anything about how much support it was going to need);
- as things developed, it became available on just about any computer and operating system, and was designed specifically so that input files (files containing markup instructions; this is not a WYSIWYG system) would be portable, and would generate the same output on any system on which they were processed – same hyphenations, line breaks, page breaks, etc., etc.;
- other programs available at the time for mathematical composition were:
  - ⋆ proprietary
  - ⋆ very expensive
  - ⋆ often limited to specific hardware
  - ⋆ if WYSIWYG, the same expression in two places in the same document might very well not look the same, never mind look the same if processed on two different systems.

Mathematicians are traditionally, shall we say, frugal; their budgets have not been large (before computer algebra systems, pencils, paper, chalk and blackboards were the most important research tools). TEX came along just before the beginnings of the personal computer; although it was developed on one of the last of the 'academic' mainframes (the DECsystem (Edusystem)-10 and -20), it was very quickly ported to some early HP workstations and, as they emerged, the new personal systems. From the start, it has been popular among mathematicians, physicists, astrophysicists, astronomers, any research scientists who were plagued by lack of the necessary symbols on typewriters and who wanted a more professional look to their preprints.

To produce his own books, Knuth had to tackle all the paraphernalia of academic publishing—footnotes, floating insertions (figures and tables), etc. As a mathematician/computer scientist, he developed an input language that makes sense to other scientists, and for math expressions, is quite similar to how one mathematician would recite a string of notation to another on the telephone. The TeX language is an interpreter. It accepts mixed commands and data. The command language is very low level (skip so much space, change to font X, set this string of words in paragraph form, . . .), but is amenable to being enhanced by defining macro commands to build a very high level user interface (this is the title, this is the author, use them to set a title page according to AMS specifications). The handling of footnotes and similar structures are so well behaved that 'style files' have been created for TeX to process critical editions and legal tomes. It is also (after some highly useful enhancements in about 1990) able to handle the composition of many different languages according to their own traditional rules, and is for this reason (as well as for the low cost), quite widely used in eastern Europe.

Some of the algorithms in TeX have not been bettered in any of the composition tools devised in the years since TeX appeared. The most obvious example is the paragraph breaking: text is considered a full paragraph at a time, not line-by-line; this is the basic starting algorithm used in the HZ-program by Peter Karow (and named for Hermann Zapf, who developed the special fonts this program needs to improve on the basics).

In summary, TeX is a special-purpose programming language that is the centerpiece of a typesetting system that produces publication quality mathematics (and surrounding text), available to and usable by individuals.

1.3 What is LaTeX then?

At the beginning of 1980s, Leslie Lamport started work on a document preparation system called LaTeX based on the TeX formatter. This system adds a set of functions that makes the TeX language more friendlier than using the primitives provided in TeX, enabling the author to concentrate on the content and structure of the document rather than the formatting details, so that the author might not loose the train of thought while writing his document. Also, LaTeX's functionality, in conjunction with a few auxiliary programs, includes the generation of indices, bibliographies, cross-references, tables of contents, graphic inclusion, etc. These are the features that are lacking in basic TeX (usually called plain TeX).

1.4 Getting Started

First of all, let's see what steps are necessary to produce a document using LaTeX. The first step is to type the file that LaTeX reads. This is usually called the LaTeX file or the input file, and it can be created using a simple text editor (in fact, if you're using a fancy word processor, you have to be sure that your file is saved in ASCII or non-document mode without any special control characters). The LaTeX program then reads your input file and produces what is called a DVI file (DVI stands for DeVice Independent). This file is not readable, at least not by humans. The DVI file is then read by another program (called a device driver) that produces the output that is readable by humans. Why the extra file? The same DVI file can be read by different device drivers to produce output on a dot matrix printer, a laser printer, a screen viewer, or a phototypesetter. Once you have produced a DVI file that gives the right output on, say, a screen viewer, you can be assured that you will get exactly the same output on a laser printer without running the LaTeX program again.

The process may be thought of as given in the Figure 1.1. This means that we don't see our output in its final form when it is being typed at the terminal. But in this case a little patience is amply rewarded, for a large number of symbols not available in most word processing programs become available. In addition, the typesetting is done with more precision, and the input files are easily sent between different computers by electronic mail or on a magnetic medium.

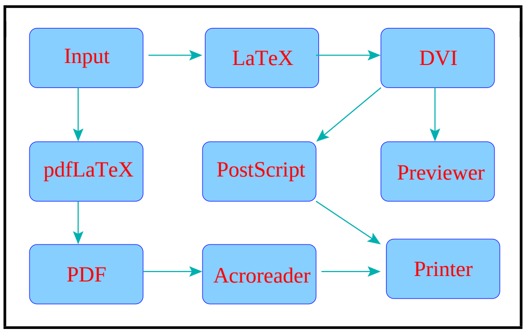


Figure 1.1 The LaTeX production chain

```
\documentclass[a4paper]{tutorial}
\pagestyle{headings}
\usepackage[screen,rightpanel,paneltoc,code]{pdfscreen}

\begin{document}

\chapter{Introduction}

\section{The Concept of Generic Markup}
Originally markup was the annotation of manuscripts
of a copy editor telling the typesetter how to format
the manuscript. It consisted of handwritten notes such
as ‘\emph{set this heading in 12 point Helvetica}
```

Figure 1.2 A sample LaTeX input file

Our focus will be on the first step, that is, creating the LaTeX input file and then running the LaTeX program to produce appropriate results. There are two ways of running the LaTeX program; it can be run in batch mode or interactively. In batch mode you submit your LaTeX input file to your computer; it then runs the LaTeX program without further intervention and gives you the result when it is finished. In interactive mode the program can stop and get further input from the user, that is, the user can interact with the program. Using LaTeX interactively allows some errors to be corrected by the user, while the LaTeX program makes the corrections in batch mode as best it can. Interactive is the preferred mode, of course. All personal computer and many mainframe implementations are interactive. On some mainframes, however, the only practical method of running LaTeX is in batch mode.

1.4.1 A typical LaTeX input file

The preamble portion of a LaTeX input file that generated the *Introduction* page of this document is given in Figure 1.2. You will notice that there are many keywords that start with the character ‘\’ followed by arguments within ‘`[ ]`’ and ‘`{ }`’. These keywords are called *control sequences*, the arguments within square brackets are called optional arguments and those within curly braces are called arguments (which are mandatory). We will learn about these later on.

When you run LaTeX over this file (for the time being, we shall name it as `test.tex`), we get the output called `test.dvi`. The `web2c` TeX system is the implementation distributed by the TeX Users Group and is free. Throughout this tutorial, we shall describe TeX functionality based on `web2c` system only. Commercial implementations like PCTeX and Y&YTeX for Win32 systems or *Textures* for Macintosh, though widely used in the typesetting industry, will not be described in this manual owing to its non-GNU nature.

You can issue the following command to the command prompt of your Unix shell to compile your input file (here we call `test.tex`):

```
$ latex test
```

Extension is only necessary, if you have given extension other than `*.tex`. In Win32 system, you can use the TEXshell and can click at the LaTeX button to run LaTeX.

Many previewers are available, `xdvi` is the standard previewer in Unix and `Windvi` in Win32 systems. The following command will show your dvi in your computer screen. Again, extension is only optional.

```
$ xdvi test
```

Printing is usually done through POSTSCRIPT. You can convert the `dvi` into `ps` by issuing the following command:

```
$ dvips test -o test.ps
$ lpr test.ps
```

This will print the dvi to your printer. `dvips` can be configured to pipe the `*.ps` directly to your printer. Win32 systems provide menu buttons to accomplish these jobs.

You might be amused to know that this `test.dvi` is independent of any platform and devices. You can view this output in any dvi previewer of any operating system irrespective of the OS of origination and can be printed in any printer for the identical output, which is not the case with the WYSIWYG typesetting systems that are usually hard wired to the installed printer, the format changes as soon as you change your printer. Therefore, TeX is device and platform independent. Also you can compile the very same TeX sources in any TeX system in any operating system irrespective of its originating OS. This platform independence has made TeX documents a choice of transfer, especially scientific documents over the INTERNET.

2 Some Conventions

As a well developed programming language, TEX has certain conventions that might be worth understanding. It might appear cryptic to learn such nitty-gritty things just to typeset a document, but it will eventually become known to you that it is worth understanding. As in other mainstream programming languages, TEX has data types, booleans, input/output operations, etc. Apart from this TEX has a highly structured directory tree popularly called TEX Directory Structure (TDS), a font setup that is specific to TEX alone, a mechanism of reading and digesting characters that come across on its way and not found in other languages, etc. We shall examine one by one.

2.1 TEX Directory Structure

All implementation-dependent TEX system files (`.pool`, `.fmt`, `.base`, `.mem`) are stored by default directly in `texmf/web2c`. The configuration file `texmf.cnf` and various subsidiary `MakeTeX`... scripts used as subroutines are also stored there.

Non-TEX specific files are stored following the GNU coding standards. Given a root directory prefix (`/usr/local` by default), we have default locations as follows:

⟨*prefix*⟩/	installation root (`/usr/local by default)`
├── `bin/`	executables
├── `man/`	man pages
├── `info/`	info files
├── `lib/`	libraries (`libkpathsea.*`)
└── `share/`	architecture-independent files
├── `texmf/`	TDS root
└── `web2c/`	implementation-dependent files (`.pool`, `.fmt`, `texmf.cnf`, etc.)

See `ftp://ftp.gnu.org/pub/gnu/standards.text` for the rationale behind and descriptions of this arrangement. A site may of course override these defaults; for example, it may put everything under a single directory such as `/usr/local/texmf`.

2.1.1 A skeleton of a TDS

This is not to imply these are the only entries allowed. For example, local may occur at any level. Given below is the standard setup followed in `web2c` TEX implementations distributed along with GNU operating systems.

`texmf`	Top level TEX directory
├── `bibtex/`	BibTEX input files
│ ├── `bib/`	BibTEX databases
│ │ ├── `base/`	base distribution (e.g., `xampl.bib`)
│ │ └── `misc/`	single-file databases
│ ├── ⟨*package*⟩/	name of a package
│ ├── `bst/`	BibTEX style files
│ │ ├── `base/`	base distribution (e.g., `plain.bst`, `acm.bst`)

│ │ ├─misc/	single-file styles
│ └─⟨*package*⟩/	name of a package
├─doc/	Documentation
├─etex/	as with TEX, below
├─fonts/	font-related files
│ └─⟨*type*⟩/	file type (e.g., `pk`)
│ └─⟨*mode*⟩/	type of output device (for `pk` and `gf` only)
│ └─⟨*supplier*⟩/	name of a font supplier (e.g., public)
│ └─⟨*typeface*⟩/	name of a typeface (e.g., `cm`)
│ └─dpi⟨*nnn*⟩/	font resolution (for `pk` and `gf` only)
├─⟨*implementation*⟩/	TEX implementations, by name (e.g., emTEX)
├─local/	files created or modified at the local site
├─metafont/	METAFONT (non-font) input files
│ ├─base/	base distribution (e.g., `plain.mf`)
│ ├─misc/	single-file packages (e.g., `modes.mf`)
│ └─⟨*package*⟩/	name of a package (e.g., `mfpic`)
├─metapost/	MetaPost input and support files
│ ├─base/	base distribution (e.g., `plain.mp`)
│ ├─misc/	single-file packages
│ ├─⟨*package*⟩/	name of a package
│ └─support/	support files for MetaPost-related utilities
├─mft/	MFT inputs (e.g., `plain.mft`)
├─⟨*program*⟩/	TEX-related programs, by name (e.g., `dvips`)
├─source/	program source code by name (e.g., LATEX, `web2c`)
├─tex/	TEX input files
│ ├─⟨*format*⟩/	name of a format (e.g., `latex`)
│ │ ├─base/	base distribution for format (e.g., `latex.ltx`)
│ │ ├─misc/	single-file packages (e.g., `webmac.tex`)
│ │ ├─local/	local additions to or local configuration files for format
│ │ └─⟨*package*⟩/	name of a package (e.g., `graphics`, `mfnfss`)
│ ├─generic/	format-independent packages
│ │ ├─hyphen/	hyphenation patterns (e.g., `hyphen.tex`)
│ │ ├─images/	image input files (e.g., Encapsulated PostScript)
│ │ └─misc/	single-file format-independent packages (e.g., `null.tex`).

Understanding this directory tree will help you to install third party fonts and other packages later on. The top level ⟨*prefix*⟩ can be specified as is the case with TEXLive CDROM. However, the TEX tree does not change. The latest TDS Working Group's draft release is available at: `http://tug.org/tds/`.

2.2 Fonts

Unlike other typesetting systems, TEX needs a font file format called, `*.tfm` (TEX Font Metric). This file keeps all the metrics of characters like height, depth, width, kern values, etc. (TEX keeps around 64 parameters for a character) of a particular font family. During compilation process, TEX reads this metrics information and based on these values, it horizontally packs boxes having values of bounding box of characters consecutively till the end of the line is encountered. During this compilation process, TEX is not at all bothered, whether a physical font file (like `*.pfb` or `*.ttf` etc.) is available in your system. It is during previewing or printing, the corresponding software needs these font files.

It arises a problem, when one tries to access third party fonts supplied by various foundries, for instance, Adobe. Foundries do not supply TEX font metric file. However, this can easily be generated with `afm2tfm` program supplied with your TEX distribution and is a trivial process. We will learn about these in subsequent chapters.

2.3 Characters

Not all the characters of your document is seen by TEX in the same way as we see them. The following characters have special meaning, \, #, $, %, ^, &, _, {, }.

\	escape character, TEX functions or control sequences start with this character, e.g., `\alpha`, `\section`, `\bf`, etc.
#	parameter character used in TEX macros (we will learn this later on)
$	math shift character, i.e., $ character starts math mode and the next $ character stops it
%	comment character, TEX will ignore the characters after % till the end of that line
^	superscript character in math, e.g., `$a^2$` $\Rightarrow a^2$
_	subscript character in math, e.g., `$a_2$` $\Rightarrow a_2$
{	group open character used to open a local group
}	group close character used to close a local group
~	unbreakable space

The obvious question arises, what will we do if we want the above characters got printed. The table below will show you how to accomplish it:

Character	Math mode	Text mode
\	`\backslash`	`\textslash`
#	`\#`	`\#`
$	`\$`	`\$`
%	`\%`	`\%`
^	`\^`	`\^`
_	`\_`	`\_`
{	`\{`	`\{`
}	`\}`	`\}`
~	`\tilde`	`\texttilde`

2.3.1 Alphabets and numerals

Ordinary alphabets, numerals, punctuations, parentheses, square brackets, and characters other than what listed above are entered as in any other program or word processor and the result will exactly match what you have entered.

2.3.2 Mathematical symbols and notations

Greek letters, various math operators including negated operators, arrows, stretchy delimiters, etc., which are normally not available in a keyboard are entered to the computer with a set of special control sequences specifically designed for this purpose. There are around 2500 control sequences available, at least half of them are not in regular use. The numbers need not make you awe-struck, since you know most of them. Knuth designed all the control sequences in such a way that it is nothing but what you ordinarily pronounce in your classroom. For instance, if you want a Greek alpha character entered into your document, you need to give as `\alpha`, this during compilation will give you 'α'. Given below is an equation composed of such control sequences:

$$(\alpha+\beta)^2 = \alpha^2 + \beta^2 + 2\alpha\beta \tag{2.1}$$

The following code generates the above equation which is not at all difficult for any academic to undertake.

```
\begin{equation}
  (\alpha + \beta)^2 = \alpha^2 + \beta^2 + 2\alpha\beta
\end{equation}
```

Similarly, a wide variety of symbols are accessed with names similar to what we ordinarily denote them. For instance, $\swarrow$, ψ, $\longrightarrow$, $\sum$, $\subseteq$, $\not\subseteq$ are generated with `\swarrow`, `\psi`, `\longrightarrow`, `\sum`, `\subseteq`, `\not\subseteq`. The point is that symbols in TEX is extremely logical to follow and not much extra effort is needed to understand and remember them. We will learn more about math symbols, formulae and their spatial arrangement and constructs during the second phase of our tutorial.

2.3.3 Accented characters

Languages other than English have a variety of accents and special symbols. LATEX provides commands to generate accents and symbols to put small pieces of non-English text in an English document. See this sentence:

El señor está bien, garçon, Él está aqüí

generated by the following code:

```
El se\~nor est\'a bien, gar\c{c}on, \'El est\'a aq\"u\'{\i}
```

List of commands for accents and special symbols

`` \`{o} ``	⟹	ò	`\~{o}`	⟹	õ
`\'{o}`	⟹	ó	`\={o}`	⟹	ō
`\^{o}`	⟹	ô	`\.{o}`	⟹	ȯ
`\"{o}`	⟹	ö	`\u{o}`	⟹	ŏ
`\v{o}`	⟹	ǒ	`\c{o}`	⟹	o̧
`\H{o}`	⟹	ő	`\d{o}`	⟹	ọ
`\t{oo}`	⟹	o͡o	`\b{o}`	⟹	o̲
`\oe`	⟹	œ	`\aa`	⟹	å
`\OE`	⟹	Œ	`\AA`	⟹	Å
`\ae`	⟹	æ	`\AE`	⟹	Æ
`\o`	⟹	ø	`\O`	⟹	Ø
`\l`	⟹	ł	`\L`	⟹	Ł
`\ss`	⟹	ß			
`\dag`	⟹	†	`\ddag`	⟹	‡
`\S`	⟹	§	`\P`	⟹	¶
`\copyright`	⟹	©	`\pounds`	⟹	£

2.4 Epilog

With this chapter, we conclude the preliminaries and introductory part of the tutorial. Next chapter onwards, we get into the real meat of the learning process. The chapters have been written not from a programmer's point of view, but rather a qualitative treatment of the language from a functional point of view is undertaken. In case, any one needs any theoretical explanation of any of the functions described or its underlying mechanism in a TEX run to accomplish it, you are gladly welcomed to querry that at appropriate time. The tutorial team is only happy to explain that in great detail. So we start the LATEX document classes in the next chapter.

3 Introduction to LaTeX

LaTeX is a macro package which enables authors to typeset and print their work with the highest typographical quality, using a predefined, professional layout. Since its introduction, it has been periodically updated and revised, like all software products. For many years now the version number has been fixed at 2ε. In an effort to re-establish a genuine, improved standard, the LaTeX3 Project was set up in 1989. And the beta version of LaTeX3 has just been released. Throughout this tutorial by LaTeX, we mean LaTeX 2ε.

3.1 Basics

As explained in the previous chapter, LaTeX is quite different from WYSIWYG (what you see is what you get) approach which most modern word processors such MS Word or Corel WordPerfect follow. With these applications, authors specify the document layout interactively while typing text into the computer. Along the way they can see on the screen how the final work will look like when it is printed.

When using LaTeX it is normally not possible to see the final output while typing the text. But the final output can be previewed on the screen after processing the file with LaTeX.

Following is the method to create a LaTeX document.

(1) Type in the text with necessary commands.

(2) Compile the text with LaTeX engine.

(3) After successful compilation of the document output can be previewed on the screen.

3.2 LaTeX Input Files

The input for LaTeX is a plain ASCII text file. You can create it with any text editor. It contains the text of the document as well as the commands which tells LaTeX how to typeset the text. The commands start with a '\' (backslash character).

Eg: `\bf`, `\it`, etc.

3.2.1 LaTeX input file structure

When LaTeX2ϵ process an input file, it requires us to follow a certain structure. Thus every input file must start with the command

```
\documentclass{class}
```

When all the set up work is done, you start the body of the text with the command

```
\begin{document}
```

Now you enter the text mixed with some useful LaTeX commands. At the end of the document you add the following command

```
\end{document}
```

which tells LaTeX the end of the file. Any thing which follows this command will be ignored by LaTeX.

3.2.2 Preamble

The first command in any LaTeX file normally determines the global processing format for the entire document. The syntax for this command:

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

The possible values of class, of which one and only one may be given, are: article, book, report or letter.

The options available allow various modifications to be made to the formatting, like selecting font size — 10pt, 11pt, 12pt, specifying paper size — letterpaper, legalpaper, executivepaper, page formats — onecolumn, twocolumn etc.

The standard LaTeX class used for ordinary documents is the report class. The article class is generally used for shorter documents than the report class. book class is for real books and letter class for formatting letters.

Preamble is the portion between `\documentclass` and `\begin{`*document*`}`. This can contain package loading command like `\usepackage{`⟨*packagename*⟩`}`. Any number of `\usepackage` command can be issued or alternatively you can give the packagenames as a comma separated list in a single `\usepackage` command.

Preamble can also contain the header/footer style chosen, the command for which take the following form:

```
\pagestyle{⟨style option⟩}
```

The style options available are `empty` (header and footer empty), `plain` (page number in the footer alone, no header), `headings` (chapter heading in odd header and section heading in even header, no footers), `myheadings` (user defined text in odd and even headers, no footers). You can also define your own custom headers and footers with fancied text, boxes, graphic elements, etc.

A typical preamble of a LaTeX document will look like:

```
\documentclass[a4paper,11pt,twocolumn]{article}
\usepackage{amsmath,times}
\pagestyle{headings}
\begin{document}
```

3.3 The Document

A LaTeX document has broadly three parts *viz.*, frontmatter, mainmatter and backmatter.

3.3.1 Frontmatter

As the name specifies the frontmatter of an article has the title of the article, its authors, affiliations and an optional date which can be produced with the following commands.

```
\title{Title text}
\author{Author names and addresses}
\date{Date text}
\maketitle
```

The `\maketitle` command will trigger the typesetting of the frontmatter part.

3.3.1.1 Abstract

Abstract is produced with the command:

```
\begin{abstract}
Text for the abstract
\end{abstract}
```

In document class report, the abstract appears on a separate page without a page number; in article, it comes after the title heading on the first page. An abstract is not possible in document class book.

3.3.2 Mainmatter

This portion is the body of the document. In the case of book class this contains `\chapter`, `\section`, `\subsection` and so on. In the case of article class the rest except the `\chapter` command will appear.

3.3.3 Backmatter

Backmatter is the portion where the References, or bibliography, containing the names of other works that are referred to within the text appear.

3.4 Sectioning commands and its logical relations

The following commands are available for producing automatic, sequential sectioning:

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

With the exception of `\part`, these commands form a sectioning hierarchy. In document classes book and report, the highest sectioning level is `\chapter`. The chapters are divided into sections using the `\section` command, which are further subdivided by means of `\subsection`, and so on. In document class article, the hierarchy begins with `\section` since `\chapter` is not available.

3.4.1 Sample body of article

```
\documentclass{article}
\usepackage{amsmath}

\begin{document}

\title[Short title]{This is the title}
\author{Author, Affiliation}
\date{}

\maketitle

\begin{abstract}
This is sample abstract.
\end{abstract}

\section{Introduction}
 This is sample section.

\subsection{Subsection}
 This is sample subsection.

\subsubsection{Subsubsection}
```

This is sample subsubsection.

`\paragraph{`*Paragraph*`}`
This is sample paragraph.

`\begin{`*thebibliography*`}{`*00*`}`

`\bibitem{`*1*`}` This is sample bibitem one.
`\bibitem{`*2*`}` This is sample bibitem two.
`\bibitem{`*3*`}` This is sample bibitem three.

`\end{`*thebibliography*`}`

`\end{`*document*`}`

3.4.2 Numbering of heading levels

In LaTeX the numbering of heading levels are automatically taken care of by LaTeX. The default numbering is Arabic. In article class, the top level sectioning unit is `\section` which will start with Arabic '1' and subsequent sections will increment by one. `\subsection` will start with Arabic number '1.1' (*ie.*, section-no.subsection-no) and next subsection will be 1.2 but the subsection counter will reset automatically when next `\section` command is encountered. Similarly, all the sectioning units behave within its heirarchial order.

If you want to avoid the number of a particular heading level, you might add `*` character to the command, which is called a *starred version* of that command. The usual convention in LaTeX is that all the starred versions do not have counter numbers. An example of a `\section` heading without number will be:

```
\section*{⟨some heading⟩}
```

3.5 Notes

The above discussion is not exhaustive. Readers might be left with many doubts in formatting the various heading levels, different types of numbers, formatting numbers, alignment, custom page-styles, etc. Queries concerning the above are invited from subscribers which will be answered in detail and will be added to this chapter as FAQ.

Even otherwise, we will revisit the documentclass issues little later, when we shall discuss the advanced features that we consciously missed this time.

4 Lists, etc.

4.1 Lists

There are three list environments available for producing formatted lists:

`\begin{`*enumerate*`}`	*list text*	`\end{`*enumerate*`}`
`\begin{`*itemize*`}`	*list text*	`\end{`*itemize*`}`
`\begin{`*description*`}`	*list text*	`\end{`*description*`}`

4.1.1 Sample `enumerate`

(1) The labels consists of sequential numbers.

(2) The numbers starts at 1 with every call to the enumerate environment.

```
\begin{enumerate}
\item The labels consists of sequential numbers.
\item The numbers starts at 1 with every call to the
      enumerate environment.
\end{enumerate}
```

4.1.2 Sample `itemize`

- The individual entries are indicated with a black dot, so-called bullet.
- The text in the entries may be of any length.

```
\begin{itemize}
\item The individual entries are indicated with a black dot,
      a so-called bullet.
\item The text in the entries may be of any length.
\end{itemize}
```

4.1.3 Sample `description`

Purpose: This environment is appropriate when a number of words or expressions are to be defined. This environment is appropriate when a number of words or expressions are to be defined.

Example: It may also be used as an author list in the bibliography.

```
\begin{description}
\item[Purpose:] This environment is appropriate when a number of
                words or expressions are to be defined. This
                environment is appropriate when a number of words or
                expressions are to be defined.

\item[Example:] It may also be used as an author list in the
                bibliography.
\end{description}
```

4.1.4 Nesting of lists

The above lists may be included within one another, either mixed or of one type, to a depth of four levels. An example of a nested list with mixed types:

- The `itemize` label at the first level is a bullet.
  1. (1) The numbering is with Arabic numerals since this is ...
     - ⋆ This is the third level of the nesting, but the ...
       - (a) And this is the fourth level of the overall ...
       - (b) Thus the numbering is with lower case letters ...
     - ⋆ The label at this level is a long dash.
  2. (2) Every list should contain at least two points.
- Blank lines ahead of an ...

```
\begin{itemize}
  \item The {\tt itemize} label at the first level is a bullet.
    \begin{enumerate}
      \item The numbering is with Arabic numerals since this is ...
        \begin{itemize}
          \item This is the third level of the nesting, but the ...
            \begin{enumerate}
              \item And this is the fourth level of the overall ...
              \item Thus the numbering is with lower case letters ...
            \end{enumerate}
          \item The label at this level is a long dash.
        \end{itemize}
      \item Every list should contain at least two points.
    \end{enumerate}
  \item Blank lines ahead of an ...
\end{itemize}
```

4.1.5 Manipulation of list numbers

(1) First level item
(2) First level item
    (a) Second level item
    (b) Second level item
        i. Third level item
        ii. Third level item
            A. Fourth level item
            B. Fourth level item
        iii. Third level item
        iv. Third level item
    (c) Second level item
    (d) Second level item
(3) First level item
(4) First level item

The default numbering scheme of list level 1 is Arabic numbers, level 2 is lowercase letters, level 3 is lower case Roman numeral and level 4 is uppercase letters. These numbers can be changed by redefining the commands that typeset the numbers of various list levels. `\theenumi`, `\theenumii`, `\theenumiii` and `\theenumiv` correspond to the number label in different levels of enumerated lists. `\labelenumi`, `\labelenumii`, `\labelenumiii` and `\labelenumiv` relate to the attributes of the number label in different levels of itemized lists.

If you want to change the default number scheme of the first level of enumerated list to bold uppercase Roman numeral enclosed within a pair of square brackets, you might issue the following command just before the start of the `\begin{`*enumerate*`}` command:

```
\renewcommand\theenumi{\Roman{enumi}}
\renewcommand\labelenumi{[{\bfseries\theenumi}]}
```

[**I**] First level item
[**II**] First level item

The commands `\arabic`, `\roman`, `\Roman`, `\alph`, `\Alph` will yield Arabic number (1, 2, 3, . . .), lowercase Roman numeral (i, ii, iii, . . .), uppercase Roman numeral (I, II, III, . . .), lowercase alphabet (a, b, c, . . .) and uppercase alphabet (A, B, C, . . .) respectively.

4.1.6 Manipulation of list labels

- First level item
- First level item
  – Second level item
  – Second level item
    ∗ Third level item
    ∗ Third level item
      · Fourth level item
      · Fourth level item
    ∗ Third level item
    ∗ Third level item
  – Second level item
  – Second level item
- First level item
- First level item

The default label scheme of itemized list level 1 is `\textbullet` (•), level 2 is `\textendash` (–) , level 3 is `\textasteriskcentered` (∗) and for level 4 is `\textperiodcentered` (·). These labels can be changed by redefining the commands that typeset the labels of various list levels. `\labelitemi`, `\labelitemii`, `\labelitemiii` and `\labelitemiv` correspond to the labels in different levels of itemized lists. If you want to change the default label scheme of the first level of itemized list to unfilled square, you might issue the following command just before the `\begin{`*itemize*`}` command:

```
\renewcommand\labelitemi{$\square$}
```

☐ First level item
☐ First level item

4.2 Displayed text

Quite often we might be needed to typeset text material in a different way than the ordinary sentences to highlight its importance. These are normally called *displayed text*. LaTeX provides three environments, `quote`, `quotation` and `verse` for displaying your text, the normal usage is shown below:

`\begin{`*quote*`}`	text	`\end{`*quote*`}`
`\begin{`*quotation*`}`	text	`\end{`*quotation*`}`
`\begin{`*verse*`}`	text	`\end{`*verse*`}`

A section of text will be displayed by indenting it by an equal amount on both sides, with these environments.

4.2.1 Quote and quotation

The example of `quote` environment given below is self explanatory. The left box gives you the code and right one is the typeset output.

```
... example of a short displayed
quotation.
\begin{quote}
It's a good idea to make your input
file as easy to read as possible.
\end{quote}
```

The following is an example of a short displayed quotation.

> It's a good idea to make your input file as easy to read as possible.

`quote` is limited to a single paragraph, while `quotation` can be used to display texts running to paragraphs.

4.2.2 Poetry

Poetry is displayed with the `verse` environment. A new stanza is begun with one or more blank lines; lines within a stanza are separated by `\\` command.

```
\begin{verse}
 There is an environment for verse
 Whose features some poets will curse

 For instead of making
 Them do \emph{all} line breaking,
 It allows them to put many words on a line when they'd rather be
 forced to be terse.
\end{vese}
```

The above code will generate the following output:

There is an environment for verse
Whose features some poets will curse

For instead of making
Them do *all* line breaking,
It allows them to put many words on a line when they'd
rather be forced to be terse.

5 Several Kinds of Boxes

The theory of composing pages out of boxes lies at the very heart of TeX and many LaTeX 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. LaTeX 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.

5.1 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, without or with a frame drawn around it. For instance, `\fbox{`*some words*`}` gives $\boxed{\text{some words}}$ whereas `\mbox` will do the same thing, but without the ruled frame around the text.

```
\mbox{text}
\makebox[width][pos]{text}
\fbox{text}
\framebox[width][pos]{text}
```

The commands in the third and fourth lines are a generalization of the other commands. They allow the user to specify the width of the box and the positioning of text inside.

some words

$\boxed{\hspace{4cm}\text{some words}}$

```
\makebox[5cm]{some words}      \par
\framebox[5cm][r]{some words}
```

In addition to the centering the text with positional argument `[c]` (the default), you can position the text flush left (`[l]`). LaTeX 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 LaTeX, 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`, `\depth` and `\totalheight`. They specify the natural size of the text, where `\totalheight` is the sum of the `\height` and `\depth`.

A few words of advice

A few words of advice

A few words of advice

```
\framebox{A few words of advice}
\framebox[\width+4mm][s]{A few words of advice}
\framebox[1.5\width]{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]{\color{red}$\Leftrightarrow$}
As seen in the margin of the ...
```

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

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}
```

Given below is an example of lowered and elevated text boxes.

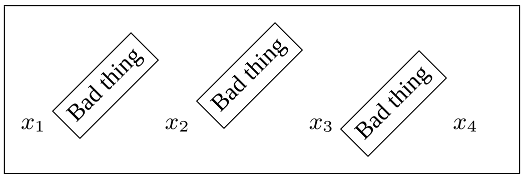
baseline $^{\text{upward}}$ baseline $_{\text{downward}}$ baseline

```
baseline \raisebox{1ex}{upward} baseline
\raisebox{-1ex}{downward} baseline
```

As with `\makebox` and `\framebox` the LaTeX 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.:



```
$x_1$ \doturn{\fbox{Bad thing}}
$x_2$ \doturn{\raisebox{\depth}
          {\fbox{Bad thing}}}
$x_3$ \doturn{\raisebox{-\height}
          {\fbox{Bad thing}}} $x_4$
```

5.2 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 *width*. The vertical positioning of the box with respect the text baseline is controlled by the one-letter optional parameter *pos* (`[c]`, `[t]`, and `[b]`).

```
\parbox[pos]{width}{text}
```

is the usage for `\parbox` command, whereas that of the `minipage` environment will look like:

```
\begin{minipage}[pos]{width}
... here goes the text matter ...
\end{minipage}
```

The center position is the default as shown by the next example. You can also observe that LaTeX might produce wide inter-word spaces if the measure is incredibly small.

This is the contents of the left-most parbox.

CURRENT LINE

This is the right-most parbox. Note that the typeset text looks sloppy because LaTeX cannot nicely balance the material in these narrow columns.

The code for generating these three `\parbox`'s in a row is given below:

```
\parbox{.3\linewidth}
 {This is the contents of the left-most parbox.}
\hfill CURRENT LINE \hfill
\parbox{.3\linewidth}
 {This is the right-most parbox. Note that the typeset
  text looks sloppy because \LaTeX{} cannot nicely balance
  the material in these narrow columns.}
```

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 indicate with a small red 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
 aligned with
\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

middle of this narrow `parbox`, which in turn is aligned with

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.

5.3 Paragraph boxes with specific height

In LaTeX, the syntax of the `\parbox` and `minipage` has been extended to include two more optional arguments.

```
\parbox[pos] [height] [inner pos]{width}{text}
```

is the usage for `\parbox` command, whereas that of the `minipage` environment will look like:

```
\begin{minipage}[pos] [height] [inner pos]{width}
 ... here goes the text matter ...
\end{minipage}
```

In both cases, *height* is a length specifying the height of the box; the parameters `\height`, `\width`, `\depth`, and `\totalheight` may be employed within the *emph* argument in the same way as in the *width* argument of `\makebox` and `\framebox`.

The optional argument *inner pos* states how the text is to be positioned internally, something that is only meaningful if *height* has been given. Its possible values are:

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

In the last case, rubber lengths (glue element) should be present where the vertical stretching is to take place.

Note the difference between the external positioning argument *pos* and the internal one *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 comprehensive.

This is a minipage with a height of 3 cm with the text aligned at the top.

In this minipage of same height, the text is vertically centered.

In this third box of same height, text is aligned at the bottom.

In this fourth box of same height, the text is stretched to fill in the entire vertical space.

See the code that generated above boxed material:

```
\fbox{%
 \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
\fbox{%
 \begin{minipage}[b][3cm][c]{2cm}
  In this minipage of same height, the text is vertically centered.
 \end{minipage}}\hfill
\fbox{%
 \begin{minipage}[b][3cm][b]{2cm}
  In this third box of same height, text is aligned at the bottom.
 \end{minipage}}\hfill
\fbox{%
 \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}}
```

5.4 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 a fbox inside a second fbox, which thus produces the double framing effect.

5.5 Rule boxes

A rule box is a basically a filled-in black rectangle. The syntax for the general command is:

```
\rule[⟨lift⟩]{⟨width⟩}{⟨height⟩}
```

which produces a solid rectangle of width *width* and height *height*, raised above the baseline by an amount *lift*. Thus `\rule`{*8mm*}{*3mm*} generates and `\rule`{*3in*}{*.2pt*} generates .

Without an optional argument *lift*, the rectangle is set on the baseline of the current line of the text. The parameters *lift*, *width* and *height* are all lengths. If *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 *height*. Such a construction is called a *strut* and is used to force a horizontal box to have a desired height or depth that is different from that of its contents.

6 Floats

6.1 Table

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.

6.1.1 Constructing tables

The environments `tabular` and `tabular*` are the basic tools with which tables can be constructed. The syntax for these environments is:

```
\begin{tabular}[pos]{cols} rows \end{tabular}
\begin{tabular*}{width}[pos]{cols} rows \end{tabular*}
```

Both the above environments actually create a minipage. The meaning of the above arguments is as follows:

pos Vertical positioning arguments (see also the explanation of this argument for parboxes). It can take on the values.

- **t** 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 to only 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:

l	the column contents are left justified
c	the column contents are centered
r	the column contents are right justified
p{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

@{text} 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, spacing is added there of an amount equal to half the normal inter-column spacing. If this spacing is not required, it may be suppressed by including an empty `@`-expression `@{}` at the beginning or end of the column format.

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 LaTeX as though they were enclosed in braces `{ }`, so that any changes 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{$n - m$}

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 begiven 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@{}rl|` contains three columns: the first is `|c@{}`, the second `r`, and the third `r|`.

6.2 Table style parameters

There are a number of style parameters used in generating tables which LaTeX 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 parameters with the command:

```
\renewcommand{\ arraystrech}{⟨factor⟩}
```

Following are the commands for changing the table style parameters that relate to dimensions:

```
\setlength\tabcolsep{⟨dimen⟩}
\setlength\arrayrulewidth{⟨dimen⟩}
\setlength\doublerulesep{⟨dimen⟩}
```

6.3 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 a row of 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.

Sample Tabular		
col head	col head	col head
Left	centered	right
aligned	items	aligned
items	items	items
Left items	centered	right aligned

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 Left       & centered & right   \\\cline{1-2}
aligned    & items    & aligned \\\cline{2-3}
items      & items    & items   \\\cline{1-2}
Left items & centered & right aligned \\
\hline
\end{tabular}
```

The discussion on tables doesn't conclude with this chapter, instead more bells and whistles are to be discussed, that includes long tables (table that spans multiple pages), how to repeat the column headings and special footlines in all multipaged tables, color tables and also few other embellishments, which the scientific community at large might require in their document preparation. So watch out for the next chapter.

6.4 Exercise

Here is an exercise you can try. If you encounter any problems, please get back to the list.

Plan for TEX Users Group 2001–2003

Project	No. ☐☐☐		Name ☐☐☐☐☐☐☐☐☐			
Year	2001		2002		2003	
	Rs.	US$	Rs.	US$	Rs.	US$
Internet costs						
Journal costs						
TEXLive production costs						
Signature				Authorization		

7 Tables Continued

7.1 Longtable

The `tabular` and `tabularx` environments explained in the previous chapter provide a convenient way of making tables. These environments, however, fail if a table exceeds one page. One easy trick that might do the job would be to break the tables manually but the `longtable` package enables automatic page breaks by the TEX compiler.

7.1.1 Constructing longtables

The `longtable` environment shares most of the features with the `tabular` environment. We begin with the following example that uses most of the features of the `longtable` environment.

Table 7.1: A long table

This part appears at the top of the table			
FIRST[1]	SECOND	*	THIRD
This table is only *slightly* different from the	one in the	*	1
guide to the `longtable`	package.	*	2
Columns 1 & 2 here have **fixed** widths.		*	3
`2.5in`	`1in`	*	
`longtable` columns are specified	in the	*	5
same way as in the `tabular`	environment.	*	6
`|p{2.5in}||p{1in}@{*}c|`	in this case.	*	7
Each line ends with a	\\ command.	*	8
The \\ command has an	optional	*	9
argument, just as	in the	*	10
`tabular`	environment.	*	11
See the effect of `\\[10pt]` below:		*	12
The `\tabularnewline` command is an alternative to \\ for use in the scope of `\raggedleft` and similar commands that redefine \\.		*	13
Notice `\tabularnewline[10pt]` below:		*	14
Some lines can be really long: This column is a “p” column so that this row of the table can take up several lines. But TEX will never break a page within such a row.	This is also a “p” column.	*	This is not.
\\* has the same effect as \\	but it	*	16
disallows a page break after	the row.	*	17
`setlongtables` is an obsolete	command in	*	18
`v4.09` of `longtable` and does	nothing.	*	19
This goes at the	bottom.	*	0.00

[1] You can also have a footnote in the table head by using `\footnotemark` and `\footnotetext`.

Table 7.1: (continued)

This part appears at the top of every other page			
First	**Second**	*	**Third**
Center aligned text	⋄	*	20
Right aligned text	◦	*	21
Also `\hline` can be used as in `tabular`.		*	22
That was a `\hline`.		*	23
That was a `\hline\hline`.		*	24
This line is produced by `\multicolumn`[2]...			
Lots of lines	like this.	*	25
Lots of lines	like this.	*	26
Lots of lines	like this.	*	27
Lots of lines	like this.	*	28
Lots of lines	like this.	*	29
One[3] line	like this.	*	30
Another one	like this[4]	*	31
Lots of lines	like this.	*	32
These lines will	appear	*	1.00
in place of the	usual foot	*	2.00
at the end	of the table.	*	3.00

7.1.2 Optional arguments

The optional arguments to `\begin`{*longtable*} are:

- **c** The table is set center aligned.
- **l** The table is set flush left.
- **r** The table is set flush right.

If no arguments are specified, the position of the table is set according to the values of `\LTleft` and `\LTright`.

7.1.3 Commands and parameters

This section contains a brief description of the commands and other parameters that may be used in the `longtable` environment.

LTchunksize This corresponds to the number of rows that TEX has to keep in memory at one time. By default this value is set to 20, but it can be set by the user; for instance, by `\setcounter{LTchunksize}{10}` or `\LTchunksize=10` to a value of 10. Changing the default does not affect page breaking. However, TEX will run faster with a large `LTchunksize`, and, on the other hand, will require more memory. The minimum value of `LTchunksize` can be set equal to `1`; however, it must be at least as large as the number of rows in each of the head or foot sections (if the table head and foot need to be set).

\LTleft The defaults in the `longtable` package are such that the tables are set flush left, but are indented by the usual paragraph indentation. `\LTleft` controls the amount of glue to the left of the table. By default this is set to `\parindent`, but can be changed according to the requirements; for instance, by `\setlength \LTleft{0pt}`.

[2] We had seen more use of `\multicolumn` in the previous chapter.
[3] This is another footnote in the body (obtained by `\footnote`).
[4] Yet another footnote.

\LTright	This parameter determines the glue to the right of the table. The default for this is `\fill`.
\LTpre	Denotes the glue before the table. The default is set to `\bigskipamount`.
\LTpost	Denotes the glue after the table. The default is again `\bigskipamount`.
\LTcapwidth	This controls the width of the parbox containing the caption. The default width is set to `4in`, but changed be changed, for instance, to a value of `2in` by using `\setlength\LTcapwidth{2in}`.
\endhead	At the start of the table, one may specify the lines that are to appear at the top of every page. This command is used in place of the last `\\`.
\endfirsthead	If the head on the first page needs to be different than on the others, then one may specify the lines to appear in a normal way followed by `\endfirsthead` (in place of the last `\\`).
\endfoot	Specifies rows to appear at the bottom of every page.
\endfirstfoot	Specifies rows to appear at the bottom of the last page. The commands `\endfirsthead` and `\endlastfoot` are useful when one wants to specify something that should logically appear in the table at the end of the `firsthead`, of at the beginning of the `lastfoot`.
\\	This is similar to the `tabular` environment and specifies the end of the row.
\\[⟨*dim*⟩]	This is also similar as in the `tabular` environment and marks the end of the row, and then adds vertical space (as shown in Table (7.1)).
*	It is the same as `\\`, but disallows a page break after the row.
\tabularnewline	It is an alternative to `\\` for use in the scope of `\raggedright` and similar commands that redefine `\\`. It can also be used with an optional argument, `\tabularnewline`[⟨*dim*⟩], so as to specify the end of the row, and then add a vertical space.
\kill	If a line is `\kill`ed, by using `\kill` rather than `\\` at the end of the line, it is used in calculating column widths, but removed from the final table.
\pagebreak	Forces a page break.
\pagebreak[⟨*val*⟩]	A ‘hint’ between 0 and 4 of the desirability of a page break. A high value indicates more desirability.
\nopagebreak	Prohibits a page break.
\nopagebreak[⟨*val*⟩]	A ‘hint’ between 0 and 4 of the undesirability of a page break.
\newpage	Forces a page break.
\caption{⟨*caption*⟩}	Caption ‘Table ?: ⟨caption⟩’, and a ‘⟨caption⟩’ entry in the list of tables.
\caption[⟨*lot*⟩]{⟨*caption*⟩}	Caption ‘Table ?: ⟨caption⟩’, and a ‘⟨lot⟩’ entry in the list of tables.
\caption[]{⟨*caption*⟩}	Caption ‘Table ?: ⟨caption⟩’, but no entry in the list of tables.
\caption*{⟨*caption*⟩}	Caption ‘⟨caption⟩’, but no entry in the list of tables.
\footnote	Used for having footnotes, but it cannot be used in the table head and foot.
\footnotemark	Footnotemark, can be used only in the table head and foot.
\footnotetext	Footnote text: for use in the table body after a `\footnotemark` has been set (should appear on the page on which the footnote is desired).

\setlongtables `setlongtables` is an obsolete command in `v4.09` of `longtable` and does nothing.

\multicolumn The `\multicolumn` command works in the same way as in the `tabular` environment. Please refer to Chapter 6 for details about this.

7.2 Another example

We will show another simple example using the `longtable` environment.

Table 7.2: A simple example

Another long table example		
First two columns		Third column
p-type		
p column	another one	1
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Continued . . .		

Table 7.2: (continued)

Another long table example (continued)		
First two columns		Third column
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
Lots of line like	this	2
The End		

7.2.1 Verbatim highlights from Table (7.2):

```
\begin{longtable}{|p{0.9in}|p{2in}|c|}
xxxxx & xxxxx & xxxxxxxxxxxxx \kill
\caption{A simple example\label{simple}}\\ \hline\hline
\multicolumn{3}{|c|}{\bf Another long table example}\\ \hline\hline
\multicolumn{2}{|c|}{First two columns} & {Third column}\\ \hline
\multicolumn{2}{|c|}{p-type} & \\ \hline\hline
\endfirsthead
\caption[]{(continued)}\\ \hline\hline
\multicolumn{3}{|c|}{\bf Another long table example (continued)}\\
\hline\hline
\multicolumn{2}{|c|}{First two columns} & {Third column}\\
\hline
\endhead
\hline
\multicolumn{3}{|c|}{\bf Continued $\ldots$}\\
\hline
\endfoot
\hline
\multicolumn{3}{|c|}{\bf The End}\\
\hline
\endlastfoot
p column & another one & 1 \\
\hline
Lots of line like & this & 2 \\
...
\end{longtable}
```

7.3 Exercise

Try making Table (7.1); it seems to be quite strange, but it contains most of the features one would need in order to get thoroughly acquainted with the `longtable` environment. It should be *quite easy* if one actually reads the table itself. Look at the spaces *carefully*.

8 Color tables in LaTeX

8.1 The \colortbl package

In the previous chapters we learnt how to construct tables in LaTeX that could span even to a number of pages. Here we shall see how to obtain color cells in tables, using David Carlisle's `colortbl` package. This package requires the `color` and `array` packages.

The `colortbl` package provides a number of commands using which one can obtain *really* colorful tables. We shall demonstrate each of these with the help of simple examples in the following sections.

8.1.1 The \columncolor command

The format for the `\columncolor` command is

```
\columncolor[⟨color model⟩]{⟨color⟩}[⟨left overhang⟩][⟨right overhang⟩]
```

Color model	It changes the current color to the argument specified until the end of the current group or the environment. The colors `black`, `white`, `red`, `green`, `blue`, `cyan`, `magenta`, and `yellow` should be predefined by any driver. Colors can also be defined by a package, as well as by the use of `\definecolor` command[1].
Color	It is an optional argument, and is like a specification to the color model given. This is particularly convenient if one wants to use a color without defining it initially.
Left overhang	It controls the width of the panel past the widest entry in the column. It is also an optional argument, and takes the value `\tabcolsep` (in `tabular`) and `\arraycolsep` (in `array`).
Right overhang	If ommited it defaults to the *left overhang*.

We have a few different tables below that will demonstrate a few possibilities using the `\columncolor` command.

one	two
three	four

```
\begin{tabular}{|l|r|}
\hline
{one} & {two} \\
{three} & {four} \\\hline
\end{tabular}
```

In the following table both the overhangs are set to `0pt`.

[1] For instance, by using the command `\definecolor{myblue}{rgb}{.8,.85,1}`, you can have the color `myblue`.

one	two
three	four

```
|>{\columncolor{khaki}[0pt]}l|
 >{\color{blue}\columncolor[gray]{.8}[0pt]}r|
```

The default overhang of `\tabcolsep` produces:

one	two
three	four

```
|>{\columncolor{khaki}}l|
 >{\color{blue}\columncolor[gray]{.8}}r|
```

It is also possible to have colors like the one below! Using `\multicolumn` it is possible to change the color of specified rows of a table.

one	two
three	four

```
\multicolumn{1}{|>{\color{blue}\columncolor[gray]%
   {0.8}}l|}{three} &
   \multicolumn{1}{|>{\columncolor{khaki}}r|}{four}
```

8.1.2 The \rowcolor command

The `\rowcolor` command is helpful in case a table is maked principally by rows. The arguments in `\rowcolor` are of the form as in `\columncolor`. Here's an example:

one	two
three	four
five	six

A `\multicolumn` command overrides the default colors for both the current row and column.

```
\begin{tabular}{|l||r|}
\hline
\rowcolor{lightturquoise} {one} & {two} \\
\rowcolor{honeydew} {three} & {four} \\
\multicolumn{1}{|>{\color{blue}\columncolor[gray]%
   {0.8}}l||}{five} &
   \multicolumn{1}{|>{\columncolor{khaki}}r|}{six}\\
\hline
\end{tabular}
```

8.2 More colors and tricks

In this section we'll see how to obtain even more colors: colored rules, colored space between two rules, and more.

- Colored rules can be easily obtained wherever desired by replacing the | with something like `!{\color{green}\vline}`.
- The above trick still leaves the spaces between || white. In order to obtain colored space one can remove the inter glue, and replace it by a colored rule. For instance,

  ```
  !{\color{green}\vline}
  @{\color{yellow}\vrule width \doublerulesep}
  !{\color{green}\vline}
  ```

  would change the color of the rules to green, and there would be another yellow rule of thickness equal to `\doublerulesep` between the two.
- `\arrayrulewidth` specifies the 'thickness' of the rules. The default is set to `0.4pt` and can be changed by using, for instance, `\setwidth\arrayrulewidth{1pt}` to a value of `1pt`.

- `\arrayrulecolor` takes the same arguments as `\color`. It can be specified at any point in the table. However, if given in the mid table it affects only the rules that are specified after that point, and any vertical rules in the table 'preamble' keep their original colors. For example, the command

  ```
  \setlength\arrayrulewidth{1pt}\arrayrulecolor{blue}
  ```

  would set the `\arrayrulewidth` to `1pt` and the rule color to blue.
- `\doublerulesep` specifies the space between the double rules.
- `\doublerulesepcolor` works in the same way as `\arrayrulecolor`, and refers to the color between double rules.
- `\minrowclearance` is used for inserting space at any desired row.

8.3 Color tables with \hhline

There are many advantages of using `\hhline` (`hhline` package) to draw horizontal rules instead of `\cline`. Firstly, `\hhline` provides more flexibility in producing the rules particularly because of the way it *interacts* with the vertical lines. Moreover, sometimes the color of the lines produced by `\cline` doesn't appear (rather it's covered up by the color panels in the following row). So it becomes more appropriate if one uses the – rule type in a `\hhline` argument.

The `\hhline` command can be used to produce a single rule, or a double rule. `\hhline` has arguments very similar to those in the 'preamble' of an `array` or `tabular`.

=	A double `hline` equal to the column width.
–	A single `hline` equal to the column width.
~	A column with no `hline`.
|	A `vline` which cuts through a double (or single) `hline`.
:	A `vline` which is broken by a double line.
#	A double hline segment between two vlines.
t	The top half of a double hline segment.
b	The bottom half of a double hline segment.
∗	`*{3}{==#}` expands to `==#==#==#`, as in the ∗-form for the preamble.

We now demonstrate an example of the `\hhline` command in the following table:

A table using hhline		
S.No.	**Col. 1**	**Col. 2**
1	abc	def
2	pqr	lmn
3	uvw	xyz
n	pqr	lmn

```
\arrayrulecolor{white}
\begin{tabular}{>{\columncolor{honeydew}}c
   >{\columncolor{honeydew}}c|
   >{\columncolor{honeydew}}c}
\multicolumn{3}{>{\columncolor{wheat}}l}
  \textbf{A table using {\sf hline}}\\
\rowcolor{white} \textbf{S.No.} & \textbf{Col. 1} &
  \textbf{Col. 2}
\arrayrulecolor{black}
\rowcolor{khaki}
{1} & {abc} & {def} \\\hhline{~--}\\
\rowcolor{lightsteelblue}
{2} & {pqr} & {lmn} \\\hhline{~--}\\
{3} & {uvw} & {xyz} \\\hhline{~--}\\
\rowcolor{white}
{n} & {pqr} & {lmn}
\end{tabular}
```

8.4 More Examples of Color Table

All these examples are taken from the TEXLive CDROM. The first example is not a table, but a horizontally packed colorboxes.

cyan (C):	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9
magenta (M):	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9
yellow (Y):	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9
black (K):	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9

```
\newcounter{Colr}
\setlength{\fboxsep}{2mm}
\begin{flushleft}
  \newcommand{\CBox}[1]{\colorbox[cmyk]{.#1,0.,0.,0.}{.#1}}
  \makebox[30mm][l]{cyan (C):}
    \whiledo{\value{Colr}<10}{\CBox{\theColr}\stepcounter{Colr}}\\
  \renewcommand{\CBox}[1]{\colorbox[cmyk]{0.,.#1,0.,0.}{.#1}}
  \setcounter{Colr}{0}\makebox[30mm][l]{magenta (M):}
    \whiledo{\value{Colr}<10}{\CBox{\theColr}\stepcounter{Colr}}\\
  \renewcommand{\CBox}[1]{\colorbox[cmyk]{0.,0.,.#1,0.}{.#1}}
  \setcounter{Colr}{0}\makebox[30mm][l]{yellow (Y):}
    \whiledo{\value{Colr}<10}{\CBox{\theColr}\stepcounter{Colr}}\\
  \renewcommand{\CBox}[1]{\colorbox[cmyk]{0.,0.,0.,.\#1}{.\#1}}
  \setcounter{Colr}{0}\makebox[30mm][l]{black (K):}
    \whiledo{\value{Colr}<10}{\CBox{\theColr}\stepcounter{Colr}}
\end{flushleft}
```

LONDON					Price
Sydney	OG4G	Thu Oct 10	Mon Oct 21 or 28	11 or 18 days	**999GBP**
		Thu Oct 17	Mon Oct 21 or 28	4 or 11 days	**999GBP**
	OG7A	Sun Oct 13	Mon Oct 21 or 28	8 or 15 days	**999GBP**
		Sun Oct 20	Mon Oct 28	8 days	**999GBP**

This is a column colored table. The same table is made row coloured in the next one.

```
\setlength{\extrarowheight}{2mm}
\setlength{\tabcolsep}{2mm}
\begin{tabular}{|l|%
  >{\columncolor{yellow}}c|c|>{\columncolor{yellow}}c|c|%
  >{\columncolor{red}\bfseries}c<{\textsc{GBP}}|}
\hline
 \multicolumn{3}{>{\columncolor{red}}l}{\color{white}\textsf{LONDON}}
&\multicolumn{3}{>{\columncolor{red}}r}{\color{white}\textsf{Price}}
\\[1pt]
\hline
Sydney  & OG4G &Thu Oct 10 &Mon Oct 21 or 28 &11 or 18 days &999\\
        &      &Thu Oct 17 &Mon Oct 21 or 28 & 4 or 11 days &999\\
        & OG7A &Sun Oct 13 &Mon Oct 21 or 28 & 8 or 15 days &999\\
        &      &Sun Oct 20 &Mon Oct 28       & 8 days       &999\\
\hline
\end{tabular}
```

Sydney	OG4G	Thu Oct 10	Mon Oct 21 or 28	11 or 18 days	999
		Thu Oct 17	Mon Oct 21 or 28	4 or 11 days	999
	OG7A	Sun Oct 13	Mon Oct 21 or 28	8 or 15 days	999
		Sun Oct 20	Mon Oct 28	8 days	999

```
\setlength{\extrarowheight}{2mm}
\begin{tabular}{|l|c|c|c|c|c|c|}
\hline
Sydney    & OG4G &Thu Oct 10 &Mon Oct 21 or 28 &11 or 18 days &999\\
\rowcolor[gray]{0.5}
          &      &Thu Oct 17 &Mon Oct 21 or 28 & 4 or 11 days &999\\
          &OG7A  &Sun Oct 13 &Mon Oct 21 or 28 & 8 or 15 days &999\\
\rowcolor[gray]{0.5}
          &      &Sun Oct 20 &Mon Oct 28       & 8 days       &999\\
\hline
\end{tabular}
```

See the rule colours have different ones in the following example.

United Kingdom	London	Thames
France	Paris	Seine
Russia	Moscow	Moskva

```
\setlength{\arrayrulewidth}{2pt}
\arrayrulecolor{green}
\begin{tabular}{|l|c|r|}
\arrayrulecolor{black}\hline
    United Kingdom & London & Thames\\
\arrayrulecolor{blue}\hline
    France         & Paris  & Seine \\
\arrayrulecolor{black}\cline{1-1}
\arrayrulecolor{red}\cline{2-3}
    Russia  & Moscow &  Moskva  \\ \hline
\end{tabular}
```

It is possible to keep some cells of a table in white while keeping the whole table in a different colour.

Table title		
Description	**Column 1**	**Column 2**
Row one	mmmmm	mmmm
Row two	mmmm	mmm
Row three	mmmmm	mmmmm
Row four	mmmmm	mmmm
Totals	mmmmm	mmmmm

```
\newcommand{\CTPanel}[1]{%
 \multicolumn{1}{>{\columncolor{white}}r|}{#1}}
\setlength\fboxsep{3mm}
\colorbox[cmyk]{.40,0,0,0}{%
\begin{tabular}{l|r|r}
\multicolumn{1}{l|}
    {\large\textbf{Table title}}\\[2mm]
\textbf{Description} & \textbf{Column 1}
             & \textbf{Column 2} \\[1mm]\hline
Row one  & \CTPanel{mmmmm} & \CTPanel{mmmm} \\\hline
Row two  &  \CTPanel{mmmm} &  \CTPanel{mmm} \\\hline
Row three& \CTPanel{mmmmm} & \CTPanel{mmmmm}\\\hline
Row four & \CTPanel{mmmmm} & \CTPanel{mmmm} \\\hline
Totals   & mmmmm & mmmmm
\end{tabular}}
```

Table title		
Description	**Column 1**	**Column 2**
Row one	mmmmm	mmmm
Row two	mmmm	mmm
Row three	mmmmm	mmmmm
Row four	mmmmm	mmmm
Totals	mmmmm	mmmmm

```
\definecolor{Blueb}{cmyk}{.40,0,0,0}
\definecolor{Blued}{cmyk}{.80,0,0,0}
\arrayrulecolor{white}
\begin{tabular}{>{\columncolor{Blued}}l
                >{\columncolor{Blued}}r|%
                >{\columncolor{Blued}}r}
  \multicolumn{3}{>{\columncolor{Blueb}}l}%
    {\large\textbf{Table title}}\\[2mm]
\rowcolor{white}
  \textbf{Description} & \textbf{Column 1}
                       & \textbf{Column 2} \\[1mm]
\arrayrulecolor{black}
\rowcolor{Blueb}
  Row one   & mmmmm & mmmm \\\hhline{~--}
  Row two   &  mmmm & mmm \\\hhline{~--}
  Row three & mmmmm & mmmmm\\\hhline{~--}
  Row four & mmmmm & mmmm\\\hhline{~--}
\rowcolor{white} Totals   & mmmmm & mmmmm
\end{tabular}
```

Table title		
Description	**Column 1**	**Column 2**
Row one	mmmmm	mmmm
Row two	mmmm	mmm
Row three	mmmmm	mmmmm
Row four	mmmmm	mmmm
Totals	mmmmm	mmmmm

```
\definecolor{Blueb}{cmyk}{.40,0,0,0}
\definecolor{Blued}{cmyk}{.80,0,0,0}
\definecolor{Bluee}{cmyk}{1.0,0,0,0}
\arrayrulecolor{black}
\setlength\arrayrulewidth{1mm}
\begin{tabular}{llrrl}
\rowcolor{Blueb}
 \qquad&\multicolumn{3}{>{\columncolor{Blueb}}l}
       {\large\textbf{Table title}}&\qquad\\[2mm]
 \rowcolor{Blued}& \textbf{Description}
                 & \textbf{Column 1}
                 & \textbf{Column 2}& \\[2mm]
\hline
 \rowcolor{Blued}& Row one  & mmmmm & mmmm &\\
 \rowcolor{Blued}& Row two  &  mmmm & mmm  &\\
 \rowcolor{Blued}& Row three& mmmmm & mmmmm&\\
 \rowcolor{Blued}& Row four & mmmmm & mmmm &\\
\cline{2-3}
 \rowcolor{Bluee}& Totals & mmmmm & mmmmm&\\[2mm]
\end{tabular}
```

9 The Figure Environment

Figures are really problematical to present in a document because they never split between pages. These leads to bad page breaks which leave blank space at the bottom of pages. For the fine-tuning of that document, typesetter has to adjust the page breaks manually.

But LaTeX provides floating figures which automatically move to suitable locations. So the positioning of figures is the duty of LaTeX.

9.1 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}
```

9.1.1 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.

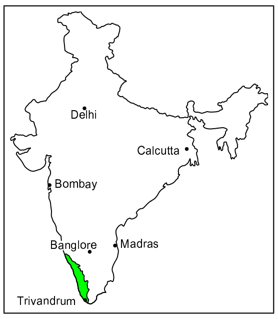


Figure 1. This is an inserted EPS graphic

- 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:

```
............ text text text text text text
\begin{figure}
.........
\end{figure}
............ text text text text text text
```

9.2 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 place that at a better location. Unprocessed floats are those which are read by LaTeX but not yet placed on the page. Though the float-placing is done by LaTeX, sometimes user has to do something 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 `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 `\section` command, inserting a `\FloatBarrier` command before each section. Note that this option is very strict. This option does not allow a float from the old 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 old section appears on the page.

`\afterpage/\clearpage`	The `afterpage` package provides the `\afterpage` command which executes a command at the next naturally-ocurring page break.

Therefore, using `\afterpage{\clearpage}` causes all unprocessed floats to be cleared at the next page break. `\afterpage{\clearpage}` is especially useful when producing small floatpage figures.

9.3 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.

9.3.1 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 `!` in the float placement options causes LaTeX to ignore these parameters. The values of these counters are set with the `\setcounter` command. For example,

```
\setcounter{totalnumber}{2}
```

prevents more than two floats from being placed on any text page.

9.3.2 Figure Fractions

The commands in the below table control what fraction of a page can be covered by floats (where “fraction” refers to the height of the floats divided by `\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 `\floatpagefraction` is always used. The value of these fractions are set by `\renewcommand`. For example,

```
\renewcommand{\textfraction}{0.3}
```

`\textfraction`	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.

`\topfraction`	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 `\textheight` from being placed at the top of a page.
`\bottomfraction`	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 `\textheight` from being placed at the bottom of a text page.
`\floatpagefraction`	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-\floatpagefraction`. The default is 0.5.

9.4 Using Graphics in LaTeX

This section shows the methods to use graphics 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 `\includegraphics`{*file.eps*}

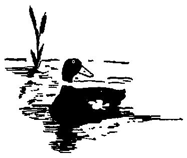
9.4.1 The \includegraphics Command

```
\includegraphics[options]{filename}
```

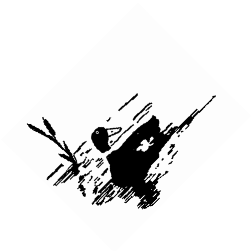
Following are the options available in `\includegraphics` command:

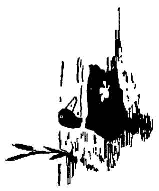
width	The width of the graphics (in any of the accepted TeX units).
height	The height of the graphics (in any of the accepted TeX units).
totalheight	The totalheight of the graphics (in any of the accepted TeX units).
scale	Scale factor for the graphic. Specifying scale=2 makes the graphic twice as large as its natural size.
angle	Specifies the angle of rotation, in degrees, with a counter-clockwise (anti-clockwise) rotation being positive.

```
\includegraphics[width=.5\textwidth]{filename}
\includegraphics[height=2in]{filename}
\includegraphics[totalheight=2in]{filename}
\includegraphics[scale=2]{filename}
```


`\includegraphics`[*width=1in*]{*duck*}

`\includegraphics`[*height=1.5in*]{*duck*}


`\includegraphics`[*scale=.25,angle=45*]{*duck*}


`\includegraphics`[*scale=.25,angle=90*]{*duck*}

9.4.2 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 `\graphicspath` command. For example,

```
\graphicspath{{dir1/}{dir2/}}
```

tells LaTeX to also look for graphics files in `dir1/` and `dir2/`. For Macintosh, this becomes

```
\graphicspath{{dir1:}{dir2:}}
```

9.4.3 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 graphic 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.eps`, then `file.ps`, then file `file.eps.gz`, etc. until a file is found. This allows the graphics to be specified with

```
\includegraphics{file}
```

instead of

```
\includegraphics{file.eps}
```

9.5 Rotating and Scaling Objects

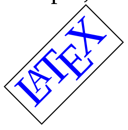
In addition to the `\includegraphics` command, the `graphicx` package includes 4 other commands which rotates and scale any LaTeX object: text, EPS graphic, etc.

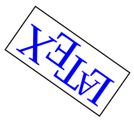
```
\scalebox{2}{\includegraphics{file.eps}}
\resizebox{4in}{!}{\includegraphics{file.eps}}
\rotatebox{45}{\includegraphics{file.eps}}
```

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 is produced with





```
\rotatebox{45}{\fbox{\Large
  \textcolor{blue}{\LaTeX}}}
```

```
\rotatebox{180}{\fbox{\Large
  \textcolor{blue}{\LaTeX}}}
```

However, the `\includegraphics` is preferred because it is faster and produces more efficient PostScript.

10 Bibliography

10.1 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. Bibliography needs consistency, LaTeX helps author to write well structured bibliography, because this is how LaTeX works—by specifying structure.

It is easy to convert the style of bibliography to publisher's require, without touching the code inside the bibliography. We can maintain a bibliographic data base using the program BibTeX. While preparing the articles, we can extract the needed references in needed style from this data base. Harvard and natbib are widely used packages for generating bibliography.

To produce bibliography, we have the environment thebibliography[1], which acts similar to enumerate environment. Here we use `\bibitem` and `\cite` commands, which do the operation similar to `\label` and `\ref`. That means in the place of citation, it will produce number or author-year code connected with list of references at the end.

```
\begin{thebibliography}[widest-label]
\bibitem{key1}
\bibitem{key2}
\end{thebibliography}
```

The `\begin{thebibliography}` command requires an argument that indicates its width, i.e., width of the widest label in the bibliography. If you know you have between 10 and 99 publications, you should start with `\begin{thebibliography}[99]`. Use any two digit number in the argument, since all numerals are the same width. If you are using customized labels, put the longest label in argument, i.e., type `\begin{thebibliography}[Long-name]`. Each entry in the environment should starts with

```
\bibitem{key1}
```

Let the author name be Alex and year 1991, the key can be coded as ale91 or else[2] as you wish. The key is used to cite publication inside the document. To cite a publication from the bibliography in the text, use the `\cite` command, which takes a key for an argument. However, the argument to `\cite` may be one key, or two or more keys, separated by commas.

```
\cite{key1}
\cite{key1,key2}
```

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 to this argument will be placed within the square brackets, after the label.

[1] Bibiliography environment need two compilation. In first compilation it will generate file with aux extension, where citation and bibcite will be marked and in second compilation \cite will be replaced by numeral or author-year code.

[2] Key can be any sequence of letters, digits and punctuation characters, except that it may not contain a comma (maximum 256 characters).

```
\cite[page 25]{key1}
```

input—file

```
It is  hard to write unstructured and disorganised documents using
\LaTeX~\cite{les85}.It is interesting to type set one
equation~\cite[Sec 3.3]{les85} rather than setting 10 pages of
running matter~\cite{don89,rondon89}.

\begin{thebibliography}{9}
\bibitem{les85}Leslie Lamport, 1985. \emph{\LaTeX---A Document
Preparation System---User's Guide and Reference Manual},
Addision-Wesley, Reading.

\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}
```

output—dvi

It is hard to write unstructured and disorganised documents using LaTeX [1]. It is interesting to type set one equation [1, Sec 3.3] rather than setting 10 pages of running matter [2,3].

Bibliography

[1] Leslie Lamport, 1985. *LaTeX—A Document Preparation System—User's Guide and Reference Manual*, Addision-Wesley, Reading.

[2] Donald E. Knuth, 1989. *Typesetting Concrete Mathematics*, TUGBoat, 10(1):31-36.

[3] Ronald L. Graham, Donald E. Knuth, and Ore Patashnik, 1989. *Concrete Mathematics: A Foundation for Computer Science*, Addison-Wesley, Reading.

10.2 natbib

The natbib package is widely used for generating bibliography, because of it's flexible interface for most of the available bibliographic styles. The natbib.sty 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.sty. To load the package:

```
\usepackage[options]{natbib}
```

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.

If we want to set references in the Nature style of citations (superscripts)

```
\documentclass{article}
\usepackage{natbib}
\citestyle{nature}
\begin{document}
. . . . . . .
. . . . . . .
\end{document}
```

10.2.1 Basic commands

The natbib.sty 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 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.

`\citet{ale91}`	$\Rightarrow$	Alex et al. (1991)
`\citet[chap.~4]{ale91}`	$\Rightarrow$	Alex et al. (1991, chap. 4)
`\citep{ale91}`	$\Rightarrow$	(Alex et al., 1991)
`\citep[chap.~4]{ale91}`	$\Rightarrow$	(Alex et al., 1991, chap. 4)
`\citep[see][]{ale91}`	$\Rightarrow$	(see Alex et al., 1991)
`\citep[see][chap.~4]{jon91}`	$\Rightarrow$	(see Alex et al., 1991, chap. 4)
`\citet*{ale91}`	$\Rightarrow$	Alex, Mathew, and Ravi (1991)
`\citep*{ale91}`	$\Rightarrow$	(Alex, Mathew, and Ravi, 1991)

10.2.2 Multiple citations

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

`\citet{ale91,rav92}`	$\Rightarrow$	Alex et al. (1991); Ravi et al. (1992)
`\citep{ale91,rav92}`	$\Rightarrow$	(Alex et al., 1991; Ravi et al. 1992)
`\citep{ale91,ale92}`	$\Rightarrow$	(Alex et al., 1991, 1992)
`\citep{ale91a,ale91b}`	$\Rightarrow$	(Alex et al., 1991a,b)

10.2.3 Numerical mode

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

`\citet{ale91}`	$\Rightarrow$	Alex et al. [5]
`\citet[chap.~4]{ale91}`	$\Rightarrow$	Alex et al. [5, chap. 4]
`\citep{ale91}`	$\Rightarrow$	[5]
`\citep[chap.~4]{ale91}`	$\Rightarrow$	[5, chap. 4]
`\citep[see][]{ale91}`	$\Rightarrow$	[see 5]
`\citep[see][chap.~4]{ale91}`	$\Rightarrow$	[see 5, chap. 4]
`\citep{ale91a,ale91b}`	$\Rightarrow$	[5, 12]

10.2.4 Suppressed parentheses

As an alternative form of citation, `\citealt` is the same as `\citet` 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
`\citealt*{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
`\citetext{short comm.}`	⇒	(short comm.)

The `\citetext` command allows arbitrary text to be placed in the current citation parentheses. This may be used in combination with `\citealp`.

10.2.5 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)

10.2.6 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, textual and/or parenthetical 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.

10.2.7 Selecting citation style and punctuation

Use the command `\bibpunct` with one optional and 6 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 one is wanted.

Example 1 `\bibpunct{[}{]}{,}{a}{}{;}` changes the output of
`\citep{jon90,jon91,jam92}`
into [Jones et al. 1990; 1991, James et al. 1992].

Example 2 `\bibpunct[;]{(}{)}{,}{a}{}{;}` changes the output of
`\citep[and references therein]{jon90}`
into (Jones et al. 1990; and references therein).

11 Mathematics

11.1 Introduction

TEX is at its best while producing mathematical documents. If you want to test the power of TEX, do typeset some mathematics. In the foreword of the TEX book, Knuth writes: "TEX is a new typesetting system intended for the creation of beautiful books—and especially for books that contain a lot of mathematics".

LATEX has a special mode for typesetting mathematics. Mathematical text within a paragraph (in-line) is entered between `\(` and `\)`, between `$` and `$` or between `\begin{math}` and `\end{math}`.

Normally larger mathematical equations and formula are typesetted in separate lines, in display mode. To produce this, we enclose them between `\[` and `\]`, between `$$` and `$$` or between `\begin{displaymath}` and `\end{displaymath}`. This produces formula, which are not numbered. If we want to produce equation number, we have to use equation environment.

The spacing for both in-line and displayed mathematics is completely controlled by TEX.

11.2 Maths in text

input—file

```
Using~(5.64) and the fact that the
$c_n=\langle\psi_n\vert\Psi\rangle$
and $d_n^*=\langle X\psi_n\rangle$,
the scalar product $\langle X\vert
\Psi\rangle$ can be expressed in the
way as $\langle X\vert\Psi\rangle=
\sum_nd_n^*c_n = \mathbf{d}^\dagger
\boldsymbol{\cdot}\mathbf{c}$ where
\(\mathbf{c}\) is a column vector
with elements $c_n$ and row vector
$\mathbf{d}^\dagger$ with elements
$d_n^*$. The inverse $\mathbf{A}^{-1}$
of a matrix $\mathbf{A}$ is such that
$\mathbf{AA}^{-1}=\mathbf{A}^{-1}
\mathbf{A}= \mathbf{I}$.

Where $\mathbf{I}$ is the unit matrix, elements $I_{mn}=\delta_{mn}$. For a
\emph{stationary state} $\Psi_E=\psi_E\exp(-{\rm i}Et/\hbar)$ and a
\emph{time-independent} operator $A$ it is clear that the expectation value
\begin{math}\langle\Psi_E\vert A\vert\Psi_E\rangle=\langle\psi_E\vert
A\vert\psi_E\rangle\end{math} does not depend on the time.
```

output—dvi

Using (5.64) and the fact that the $c_n = \langle\psi_n|\Psi\rangle$ and $d_n^* = \langle X\psi_n\rangle$, the scalar product $\langle X|\Psi\rangle$ can be expressed in the way as $\langle X|\Psi\rangle = \sum_n d_n^* c_n = \mathbf{d}^\dagger \boldsymbol{\cdot} \mathbf{c}$ where $\mathbf{c}$ is a column vector with elements c_n and row vector $\mathbf{d}^\dagger$ with elements d_n^*. The inverse $\mathbf{A}^{-1}$ of a matrix $\mathbf{A}$ is such that $\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$.

Where $\mathbf{I}$ is the unit matrix, elements $I_{mn} = \delta_{mn}$. For a *stationary state* $\Psi_E = \psi_E \exp(-{\rm i}Et/\hbar)$ and a *time-independent* operator A it is clear that the expectation value $\langle\Psi_E|A|\Psi_E\rangle = \langle\psi_E|A|\psi_E\rangle$ does not depend on the time.

11.3 Fraction

```
$$
   \frac{{\rm d}\varepsilon}{{\rm d}\varepsilon}\qquad
   \frac{\frac{a}{x-y}+\frac{b}{x+y}}{1+\frac{a-b}{a+b}}
$$
```

$$\frac{{\rm d}\varepsilon}{{\rm d}\varepsilon}\qquad \frac{\frac{a}{x-y}+\frac{b}{x+y}}{1+\frac{a-b}{a+b}}$$

11.4 Equation

Don't put blank lines between the dollar signs delimiting the mathematical text. TEX assumes that all the mathematical text being typeset is in one paragraph, and a blank line starts a new paragraph; consequently, this will generate an error message.

11.4.1 Equation with numbers

```
\begin{equation}
 \varphi(x,z) = z - \gamma_{10} x - \sum_{m+n\ge2} \gamma{mn} x^m z^n
\end{equation}
```

$$\varphi(x,z) = z - \gamma_{10} x - \sum_{m+n\ge2} \gamma mn x^m z^n \qquad (1)$$

11.4.2 Equation without numbers

```
\begin{displaymath}
   \left(\int_{-\infty}^{\infty} e^{-x^2}\right)
   =\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-(x^2+y^2)}dx\,dy
\end{displaymath}
```

OR

```
$$
   \left(\int_{-\infty}^{\infty} e^{-x^2}\right)
   =\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-(x^2+y^2)}dx\,dy
$$
```

$$\left(\int_{-\infty}^{\infty} e^{-x^2}\right)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-(x^2+y^2)}dx\,dy$$

```
\[
   \left(\int_{-\infty}^{\infty} e^{-x^2}\right)
   =\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-(x^2+y^2)}dx\,dy
\]
```

$$\left(\int_{-\infty}^{\infty} e^{-x^2}\right)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-(x^2+y^2)}dx\,dy$$

11.4.3 Subequations[1]

```
\begin{subequations}
\begin{equation}
   \langle\Psi_1\vert\Psi_2\rangle\equiv\int\Psi_1^*
   (\mathbf{r})\Psi_2 (\mathbf{r}){\rm d}\mathbf{r}
\end{equation}
and
\begin{equation}
   \langle\Psi_1\vert\Psi_2\rangle\equiv\Psi_1^*(\mathbf{r}_1,\ldots,
   \mathbf{r}_N)\Psi_2(\mathbf{r}_1,\ldots,\mathbf{r}_N){\rm d}
   \mathbf{r}_1\ldots{\rm d}\mathbf{r}_N.
\end{equation}
\end{subequations}
```

$$\langle\Psi_1|\Psi_2\rangle\equiv\int\Psi_1^*(\mathbf{r})\Psi_2(\mathbf{r})\mathrm{d}\mathbf{r} \tag{2a}$$

and

$$\langle\Psi_1|\Psi_2\rangle\equiv\Psi_1^*(\mathbf{r}_1,\ldots,\mathbf{r}_N)\Psi_2(\mathbf{r}_1,\ldots,\mathbf{r}_N)\mathrm{d}\mathbf{r}_1\ldots\mathrm{d}\mathbf{r}_N. \tag{2b}$$

11.4.4 Framed displayed equation

```
\begin{equation}
   \fbox{$\displaystyle\int_0^\infty f(x)\,{\rm d}x
   \approx\sum_{i=1}^nw_i{\rm e}^{x_i}f(x_i)$}
\end{equation}
```

$$\boxed{\int_0^\infty f(x)\,\mathrm{d}x\approx\sum_{i=1}^n w_i\mathrm{e}^{x_i}f(x_i)} \tag{3}$$

11.4.5 Multiline equations – Eqnarray

```
\begin{eqnarray}
   \bar\varepsilon &=& \frac{\int_0^\infty\varepsilon
   \exp(-\beta\varepsilon)\,{\rm d}\varepsilon}{\int_0^\infty
   \exp(-\beta\varepsilon)\,{\rm d}\varepsilon}\nonumber\\
&=& -\frac{{\rm d}}{{\rm d}\beta}\log\Biggl[\int_0^\infty\exp
   (-\beta\varepsilon)\,{\rm d}\varepsilon\Biggr]=\frac1\beta=kT.
\end{eqnarray}
```

$$\begin{aligned}\bar\varepsilon &= \frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}\\ &= -\frac{\mathrm{d}}{\mathrm{d}\beta}\log\Biggl[\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\Biggr]=\frac{1}{\beta}=kT.\end{aligned} \tag{4}$$

`\nonumber` is used for suppressing number.

[1] subeqn.sty package should be loaded.

11.4.6 Matrix

```
$$
   \matrix{1 & 2 & 3\cr 2 & 3 & 4\cr 3 & 4 & 5}\qquad
   \left(\matrix{1 & \cdots & 3\cr 2 & \vdots & 4\cr
                3 & \ddots & 5}\right)
$$
```

$$\begin{matrix}1 & 2 & 3\\ 2 & 3 & 4\\ 3 & 4 & 5\end{matrix}\qquad \left(\begin{matrix}1 & \cdots & 3\\ 2 & \vdots & 4\\ 3 & \ddots & 5\end{matrix}\right)$$

11.4.7 Array

```
$$
  \begin{array}{lcll}
      \Psi(x,t) &=& A({\rm e}^{{\rm i}kx}-{\rm e}^{-{\rm i}kx})
                     {\rm e}^{-{\rm i}\omega t}&\\
                 &=& D\sin kx{\rm e}^{-{\rm i}\omega t}, & D=2{\rm i}A
  \end{array}
$$
```

$$\begin{array}{lcll}\Psi(x,t) &=& A({\rm e}^{{\rm i}kx}-{\rm e}^{-{\rm i}kx}){\rm e}^{-{\rm i}\omega t}&\\ &=& D\sin kx{\rm e}^{-{\rm i}\omega t}, & D=2{\rm i}A\end{array}$$

11.4.8 Cases

```
$$
  \psi(x)=\cases{A{\rm e}^{{\rm i}kx}+B{\rm e}^{{-\rm i}kx},
                                & for $x=0$\cr
             D{\rm e}^-{\kappa x}, & for $x=0$.}
$$
```

$$\psi(x)=\begin{cases}A{\rm e}^{{\rm i}kx}+B{\rm e}^{-{\rm i}kx}, & \text{for } x=0\\ D{\rm e}^{-\kappa x}, & \text{for } x=0.\end{cases}$$

11.4.9 Stackrel

```
$$
   a\stackrel{def}{=} \alpha + \beta\quad
   \stackrel{thermo}{\longrightarrow}
$$
```

$$a\stackrel{def}{=} \alpha + \beta\quad \stackrel{thermo}{\longrightarrow}$$

11.4.10 Atop

```
$$
   \sum_{k=1 \atop k=0} \qquad
\sum_{123 \atop{234 \atop {890 \atop 456}}}
$$
```

$$\sum_{k=1 \atop k=0} \qquad \sum_{123 \atop{234 \atop {890 \atop 456}}}$$

11.4.11 Square root

```
$$
   \sqrt[n]{\frac{x^n-y^n}{1+u^{2n}}}
$$
```

$$\sqrt[n]{\frac{x^n-y^n}{1+u^{2n}}}$$

11.4.12 Choose

```
$$
   {123 \choose 456}\qquad {x^n-y^n \choose 1+u^{2n}}
$$
```

$$ {123 \choose 456}\qquad {x^n-y^n \choose 1+u^{2n}} $$

11.5 Definitions for Theorems

We should define `\newtheorem{`*thm*`}{`*Theorem*`}` etc in preamble.

```
\newtheorem{thm}{Theorem}
\begin{thm}
This is body matter for testing this environment.
\end{thm}
```

Theorem 1 *This is body matter for testing this environment.*

```
\newtheorem{rmk}{Remark}[section]
\begin{rmk}
This is body matter for testing this environment.
\end{rmk}
```

Remark 11.5.1 This is body matter for testing this environment.

```
\newtheorem{col}{Corollary}
\begin{col}[Richard, 1987]
This is body matter for testing this environment.
\end{col}
```

Corollary 1 (Richard, 1987) *This is body matter for testing this environment.*

```
\newtheorem{lem}{Lemma}[thm]
\begin{lem}
This is body matter for testing this environment.
\end{lem}
```

Lemma 1.1 *This is body matter for testing this environment.*

```
\newtheorem{exa}{Example}[lem]
\begin{exa}
This is body matter for testing this environment.
\end{exa}
```

Example 1.1.1 *This is body matter for testing this environment.*

11.6 AMS-LaTeX[2]

Following are some of the component parts of the amsmath package, available individually and can be used separately in a `\usepackage` command:

amsbsy defines the amsmath `\boldsymbol` and (poor man's bold) `\pmb` commands.

amscd defines some command for easing the generation of commutative diagrams.

amsfonts defines the `\frak` and `\Bbb` commands and set up the fonts msam (extra math symbols A), msbm (extra math symbols B, and blackboard bold), eufm (Euler Fraktur), extra sizes of cmmib (bold math italic and bold lowercase Greek), and cmbsy (bold math symbols and bold script), for use in mathematics.

[2] CTAN: /tex-archive/macros/latex/packages/amslatex

amssymb defines the names of all the math symbols available with the $\mathcal{AMS}$ fonts collection.

amstext defines the amsmath `\text` command.

11.6.1 Align environment

Align environment is used for two or more equations when vertical alignment is desired (usually binary relations such as equal signs are aligned).

```
\begin{align}
 F_{\rm fer}(k)  =& -\frac{16 x_0 ^3 t}{3\pi }\bigg( \sum_{l=1}^\infty
                    -\frac{\nu^5}{t^4 (x_0^2-l-\frac{1}{4})^3}\bigg[S
                     \bigg(\frac{\sqrt{x_0^2+l^2}}{t};2 \bigg)
                    + 2S\bigg(\frac{\nu}{t};2 \bigg)\bigg] \bigg)\\
 F_{\rm red}(t) =& -\frac{16 x_0 ^3 t}{3\pi }\sum_{l=1}^\infty
                    \bigg\{ \frac{1}{2\nu (x_0^2+l^2)^2} \nonumber\\
                 & -\frac{\nu^5}{t^4 (x_0^2-l-\frac{1}{4})^3}\bigg[S
                    \bigg( \frac{\sqrt{x_0^2+l^2}}{t};2 \bigg)
                    +2S\bigg(\frac{\nu}{t};2 \bigg)\bigg] \nonumber\\
                 & +V(x_e ,x_{\alpha})  -g \delta (x_e - x_{\alpha}) \bigg\}.
\end{align}
```

$$\begin{aligned}
F_{\rm fer}(k) &= -\frac{16 x_0^3 t}{3\pi}\bigg(\sum_{l=1}^\infty -\frac{\nu^5}{t^4 (x_0^2-l-\frac{1}{4})^3}\bigg[S\bigg(\frac{\sqrt{x_0^2+l^2}}{t};2 \bigg) + 2S\bigg(\frac{\nu}{t};2 \bigg)\bigg] \bigg) \qquad (5)\\
F_{\rm red}(t) &= -\frac{16 x_0^3 t}{3\pi}\sum_{l=1}^\infty \bigg\{ \frac{1}{2\nu (x_0^2+l^2)^2}\\
&\quad -\frac{\nu^5}{t^4 (x_0^2-l-\frac{1}{4})^3}\bigg[S\bigg(\frac{\sqrt{x_0^2+l^2}}{t};2 \bigg) +2S\bigg(\frac{\nu}{t};2 \bigg)\bigg]\\
&\quad +V(x_e, x_{\alpha}) - g \delta (x_e - x_{\alpha}) \bigg\}. \qquad (6)
\end{aligned}$$

11.6.2 Gather environment

Gather environment is used for two or more equations, but when there is no alignment desired among them each one is centered separately between the left and right margins.

```
\begin{gather}
 \frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,{\rm d}
 \varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)\,{\rm d}\varepsilon}
 \frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,{\rm d}\varepsilon}
 {\int_0^\infty\exp(-\beta\varepsilon)}\\
 \int_0^\infty\exp(-\beta\varepsilon)\,{\rm d}\exp(-\beta\varepsilon)\nonumber\\
 \frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,{\rm d}\varepsilon}
 {\int_0^\infty\exp(-\beta\varepsilon)}\\
 \int_0^\infty\exp(-\beta\varepsilon)\,{\rm d}\exp(-\beta\varepsilon)
\end{gather}
```

$$\frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}\frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)} \tag{7}$$

$$\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\exp(-\beta\varepsilon) \tag{8}$$

$$\frac{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)} \tag{9}$$

$$\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\exp(-\beta\varepsilon) \tag{10}$$

11.6.3 Alignat environment

The align environment takes up the whole width of a display. If you want to have several "align"-type structures side by side, you can use an alignat environment. It has one required argument, for specifying the number of "align" structures. For an argument of n, the number of ampersand characters per line is $2n - 1$ (one ampersand for alignment within each align structure, and ampersands to separate the align structures from one another).

```
\begin{alignat}{2}
  L_1 & = R_1 &\qquad L_2 & = R_2\\
  L_3 & = R_3 &\qquad L_4 & = R_4
\end{alignat}
```

$$\begin{alignat}{2} L_1 & = R_1 &\qquad L_2 & = R_2 \tag{11}\\ L_3 & = R_3 &\qquad L_4 & = R_4 \tag{12}\end{alignat}$$

11.6.4 Alignment Environments as Parts of Displays

There are some other equation alignment environments that do not constitute an entire display. They are self-contained units that can be used inside other formulae, or set side by side. The environment names are: aligned, gathered and alignedat. These environments take an optional argument to specify their vertical positioning with respect to the material on either side. The default alignment is centered (`[c]`), and its effect is seen in the following example.

```
\begin{equation*}
\begin{aligned}
  x^2 + y^2 & = 1\\
          x & = \sqrt{1-y^2}
\end{aligned}
\qquad
\begin{gathered}
  (a+b)^2 = a^2 + 2ab + b^2 \\
  (a+b) \cdot (a-b) = a^2 - b^2
\end{gathered}
\end{equation*}
```

$$\begin{aligned} x^2 + y^2 &= 1\\ x &= \sqrt{1-y^2} \end{aligned} \qquad \begin{gathered} (a+b)^2 = a^2 + 2ab + b^2 \\ (a+b) \cdot (a-b) = a^2 - b^2 \end{gathered}$$

The same mathematics can now be typeset using vertical alignments for the environments.

```
\begin{equation*}
\begin{aligned}[b]
  x^2 + y^2 & = 1\\
          x & = \sqrt{1-y^2}
\end{aligned}
\qquad
\begin{gathered}[t]
  (a+b)^2 = a^2 + 2ab + b^2 \\
  (a+b) \cdot (a-b) = a^2 - b^2
\end{gathered}
\end{equation*}
```

$$\begin{aligned}[b] x^2 + y^2 &= 1\\ x &= \sqrt{1-y^2} \end{aligned} \qquad \begin{gathered}[t] (a+b)^2 = a^2 + 2ab + b^2 \\ (a+b) \cdot (a-b) = a^2 - b^2 \end{gathered}$$

11.6.5 Multline environment

The multline environment is a variation of the equation environment used for equations that do not fit on a single line. The first line of a multline will be at the left margin and the last line at the right margin except for an indentation on both sides whose amount is equal to multline-gap.

```
\begin{multline}
  {\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,{\rm d}
  \varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)\,{\rm d}
  \varepsilon}{\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,
  {\rm d}\varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)}\\
  {\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,{\rm d}
  \varepsilon}{\int_0^\infty\exp(-\beta\varepsilon)\,{\rm d}
  \varepsilon}{\int_0^\infty\varepsilon}
  {\int_0^\infty\exp(-\beta\varepsilon)}
\end{multline}
```

$$\begin{multline}\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\int_0^\infty\exp(-\beta\varepsilon)\\ \int_0^\infty\varepsilon\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\int_0^\infty\exp(-\beta\varepsilon)\,\mathrm{d}\varepsilon\int_0^\infty\varepsilon\int_0^\infty\exp(-\beta\varepsilon) \tag{13}\end{multline}$$

11.6.6 Split environment

The split environment is for single equations that are too long to fit on a single line and hence must be split into multiple lines. Unlike multline, however, the split environment provides for alignment among the split lines.

```
\begin{equation}
   \begin{split}
     (a+b)^4 & = (a+b)^2(a+b)^2\\
             & = (a^2+2ab+b^2)(a^2+2ab+b^2)\\
             & = a^4+4a^3b+6a^2b^2+4ab^3+b^4
   \end{split}
\end{equation}
```

$$\begin{split}(a+b)^4 &= (a+b)^2(a+b)^2\\ &= (a^2+2ab+b^2)(a^2+2ab+b^2)\\ &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\end{split} \tag{14}$$

11.6.7 Cases

```
\begin{equation}
   P_{r-j}=
   \begin{cases}
     0 & \text{if $r-j$ is odd},\\
     r!\,(-1)^{(r-j)/2} & \text{if $r-j$ is even}.
   \end{cases}
\end{equation}
```

$$P_{r-j}=\begin{cases}0 & \text{if $r-j$ is odd},\\ r!\,(-1)^{(r-j)/2} & \text{if $r-j$ is even}.\end{cases} \tag{15}$$

11.6.8 Matrix

```
\begin{gather*}
\begin{matrix}   0 & 1\\ 1 & 0  \end{matrix}\qquad
\begin{pmatrix} 0 & -i\\ i & 0 \end{pmatrix}\qquad
\begin{bmatrix} a & b\\ c & d  \end{bmatrix}\qquad
\begin{vmatrix} 0 & 1\\ -1 & 0 \end{vmatrix}\qquad
\begin{Vmatrix} f & g\\ e & v  \end{Vmatrix}
\end{gather*}
```

$$\begin{matrix} 0 & 1\\ 1 & 0 \end{matrix}\qquad \begin{pmatrix} 0 & -i\\ i & 0 \end{pmatrix}\qquad \begin{bmatrix} a & b\\ c & d \end{bmatrix}\qquad \begin{vmatrix} 0 & 1\\ -1 & 0 \end{vmatrix}\qquad \begin{Vmatrix} f & g\\ e & v \end{Vmatrix}$$

11.6.9 substack environment

```
\begin{equation*}
  \sum_{\substack{0\leq i\leq m\\ 0jn}}
\end{equation*}
```

$$\sum_{\substack{0\leq i\leq m\\ 0>j>n}}$$

```
\begin{equation*}
   \sum^{\substack{0\leq i\leq m\\ 0jn}}
\end{equation*}
```

$$\sum^{\substack{0\leq i\leq m\\ 0>j>n}}$$

11.6.10 Commutative Diagram[3]

```
\begin{equation*}
\begin{CD}
   S_\Lambda^{\mathcal{W}}\otimes T @j> T\\
   @VVV @VV{{\rm End}P}V\\
   (S\otimes T)/I  @=   (Z\otimes T)/J
\end{CD}
\end{equation*}
```

$$\begin{tikzcd}
S_\Lambda^{\mathcal{W}}\otimes T \arrow[r, "j"] \arrow[d] & T \arrow[d, "\mathrm{End}P"] \\
(S\otimes T)/I \arrow[r, equal] & (Z\otimes T)/J
\end{tikzcd}$$

```
\begin{equation*}
\begin{CD}
 S_\Lambda^{\mathcal{W}}\otimes T @j>  T_{XF}          @xyz> T\\
 @V{{Out}p}VV                   &      &                 @AA{{\rm End}P}A\\
 (S\otimes T)/I                 @=     X_{\mathcal{F}} @fg>  (Z\otimes T)/J
\end{CD}
\end{equation*}
```

$$\begin{tikzcd}
S_\Lambda^{\mathcal{W}}\otimes T \arrow[r, "j"] \arrow[d, swap, "Outp"] & T_{XF} \arrow[r, "xyz"] & T \\
(S\otimes T)/I \arrow[r, equal] & X_{\mathcal{F}} \arrow[r, "fg"] & (Z\otimes T)/J \arrow[u, swap, "\mathrm{End}P"]
\end{tikzcd}$$

3 amscd.sty package should be loaded.

11.6.11 Binom

```
\begin{equation*}
   \binom{x}{y}
\end{equation*}
```

$$\binom{x}{y}$$

11.6.12 $\mathcal{AMS}$ symbols

`\iint`	$\iint$	`\iiint`	$\iiint$	`\iiiint`	$\iiiint$

11.7 Mathematical Symbols

11.7.1 Lowercase Greek letters

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

11.7.2 Uppercase Greek letters

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

11.7.3 Math mode accents

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

11.7.4 Binary Operation Symbols

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

[a]Not predefined in NFSS. Use the latexsym or amssymb package.

11.7.5 Relation symbols

$\leq$	`\leq`	$\geq$	`\geq`	$\equiv$	`\equiv`	$\models$	`\models`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`	$\perp$	`\perp`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`	$\mid$	`\mid`
$\ll$	`\ll`	$\gg$	`\gg`	$\asymp$	`\asymp`	$\parallel$	`\parallel`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`	$\bowtie$	`\bowtie`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`	$\Join$	`\Join`
$\sqsubset$	`\sqsubset`	$\sqsupset$	`\sqsupset`	$\neq$	`\neq`	$\smile$	`\smile`
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\doteq$	`\doteq`	$\frown$	`\frown`
$\in$	`\in`	$\ni$	`\ni`	$\notin$	`\notin`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`				

11.7.6 Arrow symbols

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

11.7.7 Miscellaneous symbols

$\ldots$	`\ldots`	$\imath$	`\imath`	$\Im$	`\Im`	$\aleph$	`\aleph`
$\prime$	`\prime`	$\flat$	`\flat`	$\ddots$	`\ddots`	$\emptyset$	`\emptyset`
$\exists$	`\exists`	$\clubsuit$	`\clubsuit`	$\hbar$	`\hbar`	$\triangle$	`\triangle`
$\Diamond$	`\Diamond`[a]	$\Re$	`\Re`	$\Box$	`\Box`[a]	$\neq$	`\neq`
$\top$	`\top`	$\vdots$	`\vdots`	ℓ	`\ell`	$\wp$	`\wp`
$\bot$	`\bot`	∞	`\infty`	$\sharp$	`\sharp`	$\spadesuit$	`\spadesuit`
$\mho$	`\mho`	$\surd$	`\surd`	$\heartsuit$	`\heartsuit`	∂	`\partial`
$\cdots$	`\cdots`	$\jmath$	`\jmath`	$\angle$	`\angle`		
$\forall$	`\forall`	$\natural$	`\natural`	∇	`\nabla`	$\diamondsuit$	`\diamondsuit`

[a]Not predefined in NFSS. Use the latexsym or amssymb package.

11.7.8 Variable-sized symbols

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

11.7.9 Delimiters

$\uparrow$	`\uparrow`	$\}$	`\}`	$\lceil$	`\lceil`
$\{$	`\{`	$\rfloor$	`\rfloor`	$/$	`/`
$\lfloor$	`\lfloor`	$\rangle$	`\rangle`	$\Downarrow$	`\Downarrow`
$\langle$	`\langle`	$\|$	`|`	$\Updownarrow$	`\Updownarrow`
$\vert$	`|`	$\downarrow$	`\downarrow`	$\rceil$	`\rceil`
$\Uparrow$	`\Uparrow`	$\updownarrow$	`\updownarrow`	$\backslash$	`\backslash`

11.7.10 LaTeX math constructs

$\widetilde{abc}$	`\widetilde{abc}`	$\widehat{abc}$	`\widehat{abc}`
$\overleftarrow{abc}$	`\overleftarrow{abc}`	$\overrightarrow{abc}$	`\overrightarrow{abc}`
$\overline{abc}$	`\overline{abc}`	$\underline{abc}$	`\underline{abc}`
$\overbrace{abc}$	`\overbrace{abc}`	$\underbrace{abc}$	`\underbrace{abc}`
$\sqrt{abc}$	`\sqrt{abc}`	$\sqrt[n]{abc}$	`\sqrt[n]{abc}`
f'	`f'`	$\frac{abc}{xyz}$	`\frac{abc}{xyz}`

11.7.11 AMS Greek and Hebrew (available with amssymb package)

$\digamma$	`\digamma`	$\varkappa$	`\varkappa`	$\beth$	`\beth`	$\daleth$	`\daleth`	$\gimel$	`\gimel`

11.7.12 AMS delimiters (available with amssymb package)

$\ulcorner$	`\ulcorner`	$\urcorner$	`\urcorner`	$\llcorner$	`\llcorner`	$\lrcorner$	`\lrcorner`

11.7.13 AMS miscellaneous (available with amssymb package)

$\hbar$	`\hbar`	$\hslash$	`\hslash`	$\vartriangle$	`\vartriangle`
$\triangledown$	`\triangledown`	$\square$	`\square`	$\lozenge$	`\lozenge`
$\circledS$	`\circledS`	$\angle$	`\angle`	$\measuredangle$	`\measuredangle`
$\nexists$	`\nexists`	$\mho$	`\mho`	$\Finv$	`\Finv`
$\Game$	`\Game`	$\Bbbk$	`\Bbbk`	$\backprime$	`\backprime`
$\varnothing$	`\varnothing`	$\blacktriangle$	`\blacktriangle`	$\blacktriangledown$	`\blacktrinagledown`
$\blacksquare$	`\blacksquare`	$\blacklozenge$	`\blacklozenge`	$\bigstar$	`\bigstar`
$\sphericalangle$	`\sphericalangle`	$\complement$	`\complement`	$\eth$	`\eth`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`		

[a]Not defined in old releases of the amssymb package; define with the `\DeclareMathSymbol` command.

11.7.14 AMS negated arrows (available with amssymb package)

$\nleftarrow$	`\nleftarrow`	$\nrightarrow$	`\nrightarrow`	$\nLeftarrow$	`\nLeftarrow`
$\nRightarrow$	`\nRightarrow`	$\nleftrightarrow$	`\nleftrightarrow`	$\nLeftrightarrow$	`\nLeftrightarrow`

11.7.15 AMS binary relations (available with amssymb package)

$\leqq$	`\leqq`	$\leqslant$	`\leqslant`	$\eqslantless$	`\eqslantless`
$\lesssim$	`\lesssim`	$\lessapprox$	`\lessapprox`	$\approxeq$	`\approxeq`
$\lessdot$	`\lessdot`	$\lll$	`\lll`	$\lessgtr$	`\lessgtr`
$\lesseqgtr$	`\lesseqgtr`	$\lesseqqgtr$	`\lesseqqgtr`	$\doteqdot$	`\doteqdot`
$\risingdotseq$	`\risingdotseq`	$\fallingdotseq$	`\fallingdotseq`	$\backsim$	`\backsim`
$\backsimeq$	`\backsimeq`	$\subseteqq$	`\subseteqq`	$\Subset$	`\Subset`
$\sqsubset$	`\sqsubset`	$\preccurlyeq$	`\preccurlyeq`	$\curlyeqprec$	`\curlyeqprec`
$\precsim$	`\precsim`	$\precapprox$	`\precapprox`	$\vartriangleleft$	`\vartriangleleft`
$\trianglelefteq$	`\trianglelefteq`	$\vDash$	`\vDash`	$\Vvdash$	`\Vvdash`
$\smallsmile$	`\smallsmile`	$\smallfrown$	`\smallfrown`	$\bumpeq$	`\bumpeq`
$\Bumpeq$	`\Bumpeq`	$\geqq$	`\geqq`	$\geqslant$	`\geqslant`
$\eqslantgtr$	`\eqslantgtr`	$\gtrsim$	`\gtrsim`	$\gtrapprox$	`\gtrapprox`
$\gtrdot$	`\gtrdot`	$\ggg$	`\ggg`	$\gtrless$	`\gtrless`
$\gtreqless$	`\gtreqless`	$\gtreqqless$	`\gtreqqless`	$\eqcirc$	`\eqcirc`
$\circeq$	`\circeq`	$\triangleq$	`\triangleq`	$\thicksim$	`\thicksim`
$\thickapprox$	`\thickapprox`	$\supseteqq$	`\supseteqq`	$\Supset$	`\Supset`
$\sqsupset$	`\sqsupset`	$\succcurlyeq$	`\succcurlyeq`	$\curlyeqsucc$	`\curlyeqsucc`
$\succsim$	`\succsim`	$\succapprox$	`\succapprox`	$\vartriangleright$	`\vartriangleright`
$\trianglerighteq$	`\trianglerighteq`	$\Vdash$	`\Vdash`	$\shortmid$	`\shortmid`
$\shortparallel$	`\shortparallel`	$\between$	`\between`	$\pitchfork$	`\pitchfork`
$\varpropto$	`\varpropto`	$\blacktriangleleft$	`\blacktriangleleft`	$\therefore$	`\therefore`
$\backepsilon$	`\backepsilon`	$\blacktriangleright$	`\blacktriangleright`	$\because$	`\because`

11.7.16 AMS binary operators (available with amssymb package)

$\dotplus$	`\dotplus`	$\smallsetminus$	`\smallsetminus`	$\Cap$	`\Cap`
$\Cup$	`\Cup`	$\barwedge$	`\barwedge`	$\veebar$	`\veebar`
$\doublebarwedge$	`\doublebarwedge`	$\boxminus$	`\boxminus`	$\boxtimes$	`\boxtimes`
$\boxdot$	`\boxdot`	$\boxplus$	`\boxplus`	$\divideontimes$	`\divideontimes`
$\ltimes$	`\ltimes`	$\rtimes$	`\rtimes`	$\leftthreetimes$	`\leftthreetimes`
$\rightthreetimes$	`\rightthreetimes`	$\curlywedge$	`\curlywedge`	$\curlyvee$	`\curlyvee`
$\circleddash$	`\circleddash`	$\circledast$	`\circledast`	$\circledcirc$	`\circledcirc`
$\centerdot$	`\centerdot`	$\intercal$	`\intercal`		

11.7.17 AMS negated binary relations (available with amssymb package)

$\nless$	`\nless`	$\nleq$	`\nleq`	$\nleqslant$	`\nleqslant`
$\nleqq$	`\nleqq`	$\lneq$	`\lneq`	$\lneqq$	`\lneqq`
$\lvertneqq$	`\lvertneqq`	$\lnsim$	`\lnsim`	$\lnapprox$	`\lnapprox`
$\nprec$	`\nprec`	$\npreceq$	`\npreceq`	$\precnsim$	`\precnsim`
$\precnapprox$	`\precnapprox`	$\nsim$	`\nsim`	$\nshortmid$	`\nshortmid`
$\nmid$	`\nmid`	$\nvdash$	`\nvdash`	$\nvDash$	`\nvDash`
$\ntriangleleft$	`\ntriangleleft`	$\ntrianglelefteq$	`\ntrianglelefteq`	$\nsubseteq$	`\nsubseteq`
$\subsetneq$	`\subsetneq`	$\varsubsetneq$	`\varsubsetneq`	$\subsetneqq$	`\subsetneqq`
$\varsubsetneqq$	`\varsubsetneqq`	$\ngtr$	`\ngtr`	$\ngeq$	`\ngeq`
$\ngeqslant$	`\ngeqslant`	$\ngeqq$	`\ngeqq`	$\gneq$	`\gneq`
$\gneqq$	`\gneqq`	$\gvertneqq$	`\gvertneqq`	$\gnsim$	`\gnsim`
$\gnapprox$	`\gnapprox`	$\nsucc$	`\nsucc`	$\nsucceq$	`\nsucceq`
$\succnsim$	`\succnsim`	$\succnapprox$	`\succnapprox`	$\ncong$	`\ncong`
$\nshortparallel$	`\nshortparallel`	$\nparallel$	`\nparallel`	$\nvDash$	`\nvDash`
$\nVDash$	`\nVDash`	$\ntriangleright$	`\ntriangleright`	$\ntrianglerighteq$	`\ntrianglerighteq`
$\nsupseteq$	`\nsupseteq`	$\nsupseteqq$	`\nsupseteqq`	$\supsetneq$	`\supsetneq`
$\varsupsetneq$	`\varsupsetneq`	$\supsetneqq$	`\supsetneqq`	$\varsupsetneqq$	`\varsupsetneqq`

11.7.18 $\mathcal{AMS}$ arrows (available with amssymb package)

$\dashrightarrow$	`\dashrightarrow`	$\dashleftarrow$	`\dashleftarrow`	$\leftleftarrows$	`\leftleftarrows`
$\leftrightarrows$	`\leftrightarrows`	$\Lleftarrow$	`\Lleftarrow`	$\twoheadleftarrow$	`\twoheadleftarrow`
$\leftarrowtail$	`\leftarrowtail`	$\looparrowleft$	`\looparrowleft`	$\leftrightharpoons$	`\leftrightharpoons`
$\curvearrowleft$	`\curvearrowleft`	$\circlearrowleft$	`\circlearrowleft`	$\Lsh$	`\Lsh`
$\upuparrows$	`\upuparrows`	$\upharpoonleft$	`\upharpoonleft`	$\downharpoonleft$	`\downharpoonleft`
$\multimap$	`\multimap`	$\leftrightsquigarrow$	`\leftrightsquigarrow`	$\rightrightarrows$	`\rightrightarrows`
$\rightleftarrows$	`\rightleftarrows`	$\rightrightarrows$	`\rightrightarrows`	$\rightleftarrows$	`\rightleftarrows`
$\twoheadrightarrow$	`\twoheadrightarrow`	$\rightarrowtail$	`\rightarrowtail`	$\looparrowright$	`\looparrowright`
$\rightleftharpoons$	`\rightleftharpoons`	$\curvearrowright$	`\curvearrowright`	$\circlearrowright$	`\circlearrowright`
$\Rsh$	`\Rsh`	$\downdownarrows$	`\downdownarrows`	$\upharpoonright$	`\upharpoonright`
$\downharpoonright$	`\downharpoonright`	$\rightsquigarrow$	`\rightsquigarrow`		

11.7.19 Log-like symbols

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

11.7.20 Double accents in math (available with amssymb package)

$\acute{\acute{A}}$	`\Acute{\Acute{A}}`	$\bar{\bar{A}}$	`\Bar{\Bar{A}}`
$\breve{\breve{A}}$	`\Breve{\Breve{A}}`	$\check{\check{A}}$	`\Check{\Check{A}}`
$\ddot{\ddot{A}}$	`\Ddot{\Ddot{A}}`	$\dot{\dot{A}}$	`\Dot{\Dot{A}}`
$\grave{\grave{A}}$	`\Grave{\Grave{A}}`	$\hat{\hat{A}}$	`\Hat{\Hat{A}}`
$\tilde{\tilde{A}}$	`\Tilde{\Tilde{A}}`	$\vec{\vec{A}}$	`\Vec{\Vec{A}}`

11.7.21 Other Styles

11.7.21.1 Caligraphic letters

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

use `\mathcal{}`

11.7.21.2 Mathbb letters

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

use `\mathbb{}`

11.7.21.3 Mathfrak letters

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

use `\mathfrak{} with amssymb package`

11.7.21.4 Math bold italic letters

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

use `\mathbi{}`

11.7.21.5 Math Sans serif letters

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

use `\mathsf{}`

11.7.21.6 Math bold letters

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

use `\mathbf{}`

11.7.22 Accents–Symbols

ó	`\’{o}`	ö	`\"{o}`	ô	`\ˆ{o}`
ò	`\‘{o}`	õ	`\˜{o}`	ō	`\={o}`
ȯ	`\.{o}`	ŏ	`\u{o}`	ő	`\H{o}`
o͡o	`\t{oo}`	o̧	`\c{o}`	ọ	`\d{o}`
o̲	`\b{o}`	Å	`\AA`	å	`\aa`
ß	`\ss`	ı	`\i`	ȷ	`\j`
ø	`\o`	s͡	`\t s`	š	`\v s`
Ø	`\O`	¶	`\P`	§	`\S`
ṣ	`\d s`	s̊	`\r s`	s̋	`\H s`

11.8 Accents and Foreign Letters

11.8.1 Printing command characters

The characters # $ ˜_ ˆ % { } are interpreted as commands. If they are to be printed as text, the character \ must precede them:

$ = `\$`   & = `\&`   % = `\%`   # = `\#`   _ = `\_` { = `\{`   } = `\}`

11.8.2 The special characters

These special characters do not exist on the computer keyboard. They can however be generated by special commands as follows:

§= `\S`   †= `\dag`   ‡= `\ddag`   ¶= `\P`   ©= `\copyright`   £= `\pounds`

11.8.3 Foreign letters

Special letters that exist in European languages other than English can also be generated with TeX. These are:

œ= `\oe`	Œ= `\OE`	æ= `\ae`	Æ= `\AE`	å= `\aa`	Å= `\AA`	¡ = `!‘`
ø= `\o`	Ø= `\O`	ł= `\l`	Ł= `\L`	ß= `\ss`	SS= `\SS`	¿ = `?‘`

11.8.4 Accents

ò = `\‘o`	ó = `\’o`	ô = `\ˆo`	ö = `\"o`	õ = `\˜o`
ō = `\=o`	ȯ = `\.o`	ŏ = `\u o`	ǒ = `\v o`	ő = `\H o`
o͡o = `\t{oo}`	o̧ = `\c o`	ọ = `\d o`	o̲ = `\b o`	o̊ = `\r o`

The last command, `\r`, is new to LaTeX 2_ε. The *o* above is given merely as an example: any letter may be used. With *i* and *j* it should be pointed out that the dot must first be removed. This is carried out by prefixing these letters with \. The command `\i` yield ı.

12 Cross References in LaTeX

12.1 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 references 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?)

12.2 Let LaTeX do it

The basic method of using cross references (see Section 12.1 for what we mean by cross referee) in LaTeX is quite simple. Suppose that somewhere in the in the second section of you article, you want to refer to the first section. You assign a *key* to the first section using the command

```
\section{⟨section name⟩}\label{⟨key⟩}
```

and at the point in the second section where the reference is to be made, you type the command

```
\ref{⟨key⟩}
```

Thus the reference “see Section 12.1 . . . ” in the first sentence of this section was produced by including the command `\label{intro}` in the command for the first section as

```
\section{Why cross references}\label{intro}
```

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, `myfile.tex` containing cross references, the reference information in an auxiliary file named `myfile.aux` and at the end of the run LaTeX prints a warning

```
LaTeX Warning: There were undefined references.

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 *key* in `\label{`*key*`}` can be any sequence letters, digits or punctuation characters, it is convenient to use some mnemonic (such as `\label{`*limcon*`}` for a section entitled "Limits and Continuity" rather than `\label{`*sec@#*?!*`}`. Also, when you make a reference, it's better to type `~\ref{`*limcon*`}` (notice the *tie*?) than `\ref{`*limcon*`}` to prevent the possibility of the reference number falling off the edge as in " . . . . see Section 12.1 for further details. . . . "

In addition to sectioning commands such as `\chapter` or `\section`, reference can also be made to an `\item` entry in an `enumerate` environment, by attaching a `\label`. For example the input

```
In the classical \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}
Statements (\ref{pre1}) and (\ref{pre2}) are
the \emph{premises} and statement (\ref{con}) is
the conclusion.
```

gives the following output

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 *premises* and statement (3) is the conclusion

You must be a bit careful about references to tables or figures (technically, "floats"). For them, the `\label` command should be given after the `\caption` command or in its argument, as in the example below

```
\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}
Two possible relations betweeen $x$ and $y$ satisfying
the data in Table\ref{tabxy} are $y=x^3$ and
$y=6x^2-11x+6$
```

produces the following output

Value of x	1	2	3
Value of y	1	8	27

Table 12.1 Observed values of x and y

Two possible relations between x and y satisfying the data in Table 12.1 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 12.2.1 for a discussion of cross references in Mathematics."

12.2.1 Cross references in Math

Mathematical documents abounds 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{thm}[subsection]`, then after typing

```
\begin{thm}\label{diffcon}
Every differentiable function is continuous
\end{thm}
```

to get

Theorem 12.2.1.1 *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 12.2.1.1 is false.

References can be made to equations as in the following examples.

Input

```
\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
```

Output

$$(x + y)^2 = x^2 + 2xy + y^2 \qquad (12.1)$$

Changing y to $-y$ in Equation (12.1) 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 parantheses 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 (12.1) 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

Input

```
\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}
```

Output

$$(x+y)^2 = x^2 + 2xy + y^2 \qquad (12.2)$$
$$(x-y)^2 = x^2 - 2xy + y^2 \qquad (12.2a)$$

12.3 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 12.1 in page 4

To avoid the tedium of typing

```
\ref{key} on page~\pageref{key}
```

every time, you can define the macro

```
\newcommand{\fullref}[1]{\ref{#1} on page~\pageref{#1}}
```

and use `\fullref` for such references. But the trouble is that at times the referred abject and the reference to it fall on the same page (with TEX 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{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 reference are typeset by TEX 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 1, `\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 occurance 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 occurence 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 case considered above can be customized by redefining the commands generating them. For the article class without the twoside option, reference to the previous page use 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 the 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 `\reftextcurrenr` 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

see the above table

if the reference occurs on the same page as Table 12.1 and

see the tableon page 4

if they fall on different pages.

12.4 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

```
\usepakage{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 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.

12.5 Lost the keys? Use `lablst.tex`

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 LaTeX on the file `lablst.tex`. In my system, I do this by first typing

```
latex lablst
```

LaTeXresponds as follows

```
*********************************
* Enter input file name
*   without the .tex extension:
*********************************

\lablstfile=
```

I type in the file name as `cref` which is the source of this document. I'm presented with another query.

```
**********************************************
* Enter document class used in file cref.tex
*   with no options or extension:
**********************************************

\lablstclass=
```

So I type `article`. I am asked

```
********************************************
* Enter packages used in file cref.tex
*   with no options or extensions:
********************************************

\lablstpackages=
```

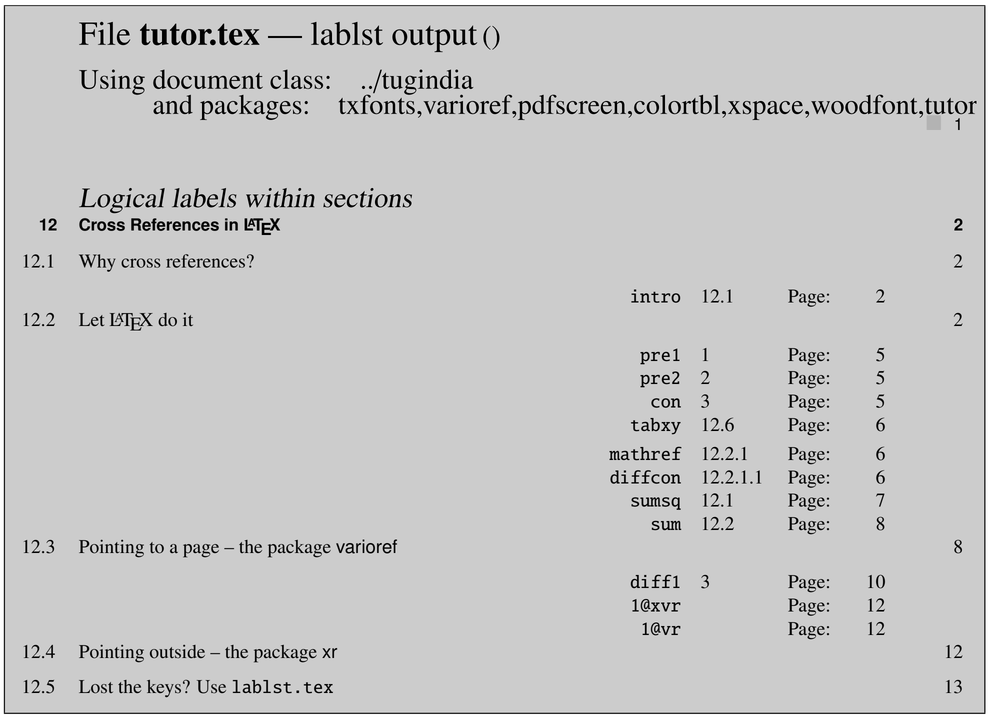
Here I need give only those packages used in the article which define commands used in section titles etc. need be given. So I type

```
amsmath,array,enumerate
```

This produces a file `lablst.dvi` which I can view 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.

Click here to see lablst output



```
        File tutor.tex — lablst output ()

        Using document class:   ../tugindia
              and packages:   txfonts,varioref,pdfscreen,colortbl,xspace,woodfont,tutor
                                                                                     1


        Logical labels within sections
  12    Cross References in LaTeX                                                    2

12.1    Why cross references?                                                        2

                                                      intro  12.1      Page:   2
12.2    Let LaTeX do it                                                              2

                                                       pre1  1         Page:   5
                                                       pre2  2         Page:   5
                                                        con  3         Page:   5
                                                      tabxy  12.6      Page:   6

                                                    mathref  12.2.1    Page:   6
                                                    diffcon  12.2.1.1  Page:   6
                                                      sumsq  12.1      Page:   7
                                                        sum  12.2      Page:   8
12.3    Pointing to a page – the package varioref                                    8

                                                      diff1  3         Page:  10
                                                      1@xvr            Page:  12
                                                       1@vr            Page:  12
12.4    Pointing outside – the package xr                                           12

12.5    Lost the keys? Use lablst.tex                                               13
```

13 A Gentle Reconnaissance

13.1 Page layout in LaTeX

A page in a LaTeX document is built from various elements as shown in figure 13.1. In a *two-sided* document, some parameters will be different for the even and odd pages. The figure shown gives the layout as on any odd page in the document. It also shows most of the parameters required in order to change the page style including the headers, footers and the margins. We shall now briefly discuss these and the other parameters that can be effectively used to control the page layout.

- The horizontal placement of the text can be set by specifying the following parameters:

\oddsidemargin	It denotes the leftside margin (on odd numbered pages). It should be noted that `\leftmargin` does not denote the leftside margin, it is instead used for the indentation of lists.
\evensidemargin	It denotes the leftside margin (on even numbered pages). Note that unless the `twoside` option is chosen, the `\oddsidemargin` and the `\evensidemargin` should be the same.
\textwidth	The width of the text.

- The parameters that control the vertical measurements are:

\topmargin	Denotes the space between the header and the vertical offset. The latter is equal to `1in + \voffset`. `1in` is the default produced by LaTeX.
\headheight	It denotes the height of the header.
\headsep	Refers to the distance between the header and the body of the text.
\textheight	Is the height of the actual text.

- The parameters that control the placement of the footer are:

\footskip	It is the distance between the body of the text and the footer.
\footheight	Denotes the height of the footer.

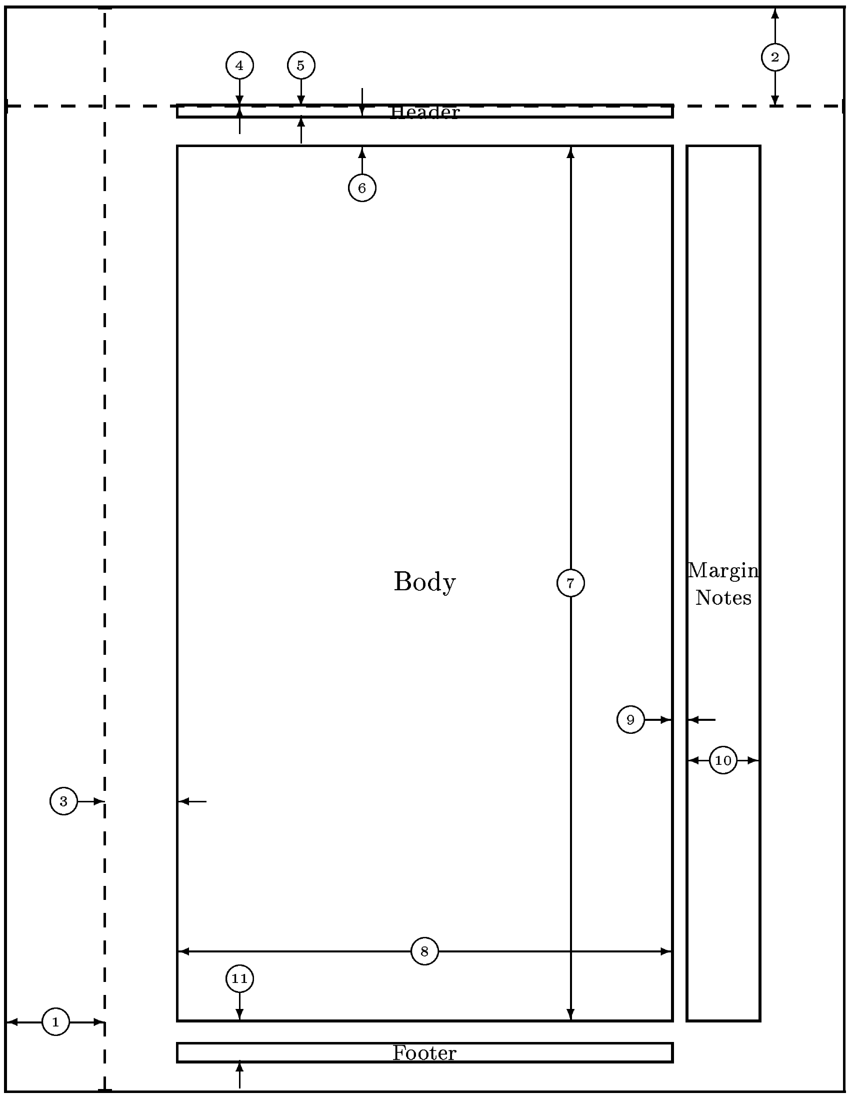
- Margin notes can be created by using the `\marginpar` command. The parameters controlling the margins are:

\marginparsep	Denotes the separation between the body of the text and the margin. It should be noted that in a two-sided document the margins appear on different sides on two consecutive pages.
\marginwidth	Denotes the width of the margin.
\marginparpush	It is the minimum vertical separation between two marginal notes.

- The commands that are needed in order to control paragraphing are:

\parskip	Denotes the vertical space between two paragraphs.
\parindent	Denotes the width of paragraph indentation.
\par	Equivalent to a blank line.
\topsep	It is extra vertical space (in addition to `\parskip`), that is added above and below list and paragraphing environments.

\itemsep It is extra vertical space (in addition to `\parskip`), that is added between two list items.



```
 1  one inch + \hoffset          2  one inch + \voffset
 3  \oddsidemargin = 54pt        4  \topmargin = 0pt
 5  \headheight = 7pt            6  \headsep = 23pt
 7  \textheight = 640pt          8  \textwidth = 362pt
 9  \marginparsep = 12pt        10  \marginparwidth = 52pt
11  \footskip = 30pt                \marginparpush = 5pt (not shown)
    \hoffset = 0pt                  \voffset = 0pt
    \paperwidth = 597pt             \paperheight = 845pt
```

Figure 13.1 Page elements. The values shown are those in effect in the current document (on odd pages), *not* the default.

The parameters defined above can be set to a particular value using the command

```
\setlength{parameter}{length}
```

Another command that can be used to change the value of a parameter by a given length is

```
\addtolength{parameter}{length}
```

13.1.1 Page headers and footers

The page headers and footers in LaTeX are defined by the `\pagestyle` and `\pagenumbering` commands. The `\pagestyle` command defines the content of the headers & footers and provides the following options:

empty	No headers or footers.
plain	No header, footer contains the page number centered. This is the default provided by LaTeX.
headings	No footer, header contains the name of the chapter/section and/or subsection and the page number.
myheadings	No footer is provided, and the header contains the page number and the information given by the `\markright` and `\markboth` commands. However, for a much better control of the headers and footers, it's recommended to use the *fancyhdr* package.

The command `\thispagestyle` can be used to change the pagestyle of the current page in the document.

The `\pagenumbering` command defines the format of the page number. The different parameters that can be used are:

arabic	roman numerals (default)
roman	lower case roman numerals
Roman	upper case roman numerals
alph	lower case letter
Alph	upper case letter `\thepage` produces the page number in the format defined by `\pagenumbering`.

13.1.2 The fancyhdr package

The `fancyhdr` package provides another parameter for specifying the pagestyle, the `fancy` style. By use of `\pagestyle{`*fancy*`}`, one can specify three-part headers and footers. We shall illustrate it's use with the help of some examples. The example below shows the page layout that can be created using the package `fancyhdr`.

LeftHeader	CenteredHeader	RightHeader
	page body	
LeftFooter	CenteredFooter	RightFooter

Here is another nice example from the `fancyhdr` documentation.

		The performance of new graduates
	page body	
From: K. Grant	To: Dean A. Smith	3

This is accomplished by the commands following `\pagestyle{fancy}`:

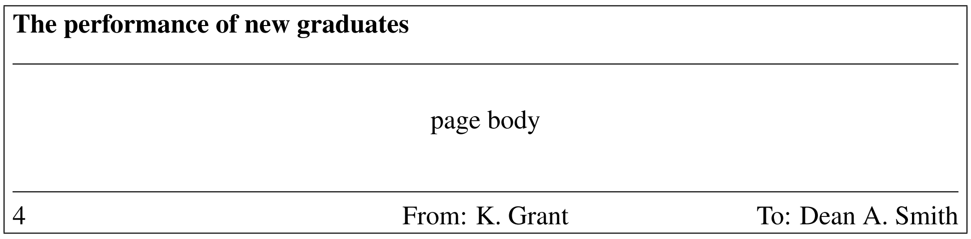
```
\lhead{}
\chead{}
\rhead{\bf The performance of new graduates}
\lfoot{From: K. Grant}
\cfoot{To: Dean A. Smith}
\rfoot{\thepage}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}
```

13.1.3 Using fancyhdr in two-sided documents

The `\fancyhdr` package also provides the commands `\fancyhead` and `\fancyfoot` which are more general than the commands described above to define the header and the footer. These provide an additional parameter that specifies for which pages and/or parts of the header/footer those apply. The selectors that can be used are:

E	Even page
O	Odd page
L	Left field
C	Center field
R	Right field
H	Header
F	Footer

Using these we can produce a two-sided document. Assuming the page layout shown above to be for the odd pages, we can have the following for the even pages:



This can be produced by using the commands:

```
\fancyhead{} % clear all fields
\fancyhead[RO,LE]{\bf The performance of new graduates}
\fancyfoot[LE,RO]{\thepage}
\fancyfoot[LO,CE]{From: K. Grant}
\fancyfoot[CO,RE]{To: Dean A. Smith}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}
```

The default layout in `fancyhdr` is produced by the following commands:

```
\fancyhead[LE,RO]{\slshape\rightmark}
\fancyhead[LO,RE]{\slshape\leftmark}
\fancyfoot[C]{\thepage}
```

The default values for `\headrulewidth` and `\footrulewidth` are `0.4pt` and `0pt` respectively.

13.2 Groups

LaTeX has an extremely nice feature of keeping text in groups thus enabling one to have different kinds of text wherever required. For example, one can have:

The available font sizes are:

tiny, scriptsize, footnotesize, small, normalsize, large, Large, LARGE, huge, and Huge.

A new group is started by the character `{` and terminated by the character `}`. It is also possible to have groups nested within groups.

> If some paragraphs need to be typeset in a different way (like this one!), then it is necessary to include `\par` or to use a blank line before closing the group, since otherwise the normal defaults will be restored before the paragraph is actually typeset.

The empty group `{}` enables one to get some space after TEX in the output. One can also print a tilde using `\~{}` (this will produce ˜). By using `\sim` in math mode, we get ∼.

And to quote the advice in **The Not So Short Introduction to LaTeX 2ε**:

Remember! *The* MO RE fonts you use **in** a `document`, *the* more READABLE and *beautiful it* becomeS.

13.3 Creating a nomenclature

In the process of writing a big document[1] which involve a number of symbols, one often feels the need to include a nomenclature for the various symbols used in the text. The `nomencl` package provides a convenient way of doing so. It makes use of the *MakeIndex* program to generate such a list automatically by using the information provided by the author in the text.

13.3.1 Package options

The `nomencl` package provides the following options:

refeq	The phrase ", see equation (⟨*eq*⟩)" is appended to every entry in the nomenclature where ⟨*eq*⟩ is the number of the last equation in front of the corresponding command `\nomenclature`.
norefeq	This is the default option; using this no equation number is printed.
refpage	The phrase ", see page (⟨*page*⟩)" is appended to every entry in the nomenclature where ⟨*page*⟩ is the number of the page on which the corresponding command `\nomenclature` appeared.
norefpage	No page reference is printed; default option.
prefix	Every sort key is preceded by the letter "a" (changeable); default option.
noprefix	No prefix is used for sorting.
cfg	A configuration file `nomencl.cfg` is loaded, if it exists; default option.
nocfg	The configuration file is not loaded.

[1] esp. mathematical documents, theses, books etc.

croatian, danish, english, french, german, italian, polish, russian, spanish, ukranian The reference texts and the nomenclature title will appear in the corresponding language. In order to use Russian or Ukranian, you'll need to have Cyrillic fonts installed and might need a replacement for *MakeIndex*, e.g. **xındy**[2]. The default option is english.

13.3.2 Usage and examples

The `\nomenclature` command has the following syntax:

```
\nomenclature[⟨perfix⟩]{⟨symbol⟩}{⟨description⟩}
```

where ⟨*prefix*⟩ is used for fine tuning the sort order, ⟨*symbol*⟩ is the symbol to be described, and ⟨*description*⟩ is the actual description. The package provides macros in order to change the referencing behavior for single entries. These macros are: `\refeq`, `\norefeq`, `\refpage`, `\norefpage`, `\refeqpage`, and `\norefeqpage`. Note that the use of these macros locally inside the command `\nomeclature` always supersedes the package options, and can be used in order to produce the desired effect. The following example will more clearly illustrate the usage of the package.

```
\documentclass{article}
\usepackage{nomencl}
\makeglossary
\renewcommand{\nomgroup}[1]{%
  \ifthenelse{\equal{#1}{A}}{\item[\textbf{Roman symbols}]}{%
    \ifthenelse{\equal{#1}{G}}{\item[\textbf{Greek symbols}]}{}}{}}
\begin{document}
\printglossary
\section{Dimensionless ratios of transport coefficients}
The {\em Lewis number} is defined as
\begin{equation}
\mathrm{Le} \equiv \frac{\lambda}{\rho C_p \mathcal{D}} =
\frac{\alpha}{\mathcal{D}} \end{equation}%
\nomenclature[ax ]{$\mathrm{Le}$}{Lewis number}%
\nomenclature[ga ]{$\lambda$}{Thermal conductivity}%
\nomenclature[ga ]{$\rho$}{Density}%
\nomenclature[a ]{$C_p$}{Constant-pressure specific heat}%
\nomenclature[g ]{$\mathcal{D}$}{Mass diffusivity}%
\nomenclature[g ]{$\alpha$}{Thermal diffusivity}%
The {\em Prandtl number} is defined as
\begin{equation}
\mathrm{Pr} \equiv \frac{C_p \mu}{\lambda} = \frac{\nu}{\alpha}
\end{equation}%
\nomenclature[ax ]{$\mathrm{Pr}$}{Prandtl number}%
\nomenclature[ga ]{$\mu$}{Dynamic viscosity}%
\nomenclature[ga ]{$\nu$}{Momentum diffusivity}%
The {\em Schmidt number} is defined as
\begin{equation}
\mathrm{Sc} \equiv \frac{\mu}{\mathcal{D}}\end{equation}%
\nomenclature[ax ]{$\mathrm{Sc}$}{Schmidt number}
\end{document}
```

As mentioned above, the `nomencl` package makes use of the MakeIndex program in or-

[2] For more information on **xındy**, please see http://gemini.iti.informatik.tu-darmstadt.de/xindy/ or http://sourceforge.net/projects/xindy/.

der to produce the nomenclature list. On running the file through LaTeX, the command `\makeglossary` instructs it to open the glossary file ⟨*jobname*⟩`.glo` corresponding to the LaTeX file ⟨*jobname*⟩`.tex` and writes the information from the `\nomenclature` commands to this file. The next step is to invoke MakeIndex in order to produce the ⟨*jobname*⟩`.gls` file. This can be achieved by making use of the command:

```
makeindex ⟨jobname⟩.glo -s nomencl.ist -o ⟨jobname⟩.gls
```

The next step is to invoke LaTeX on the file ⟨*jobname*⟩`.tex` once more. This will input the `.gls` file and process it according to the given options.

The code given in the above example produces the following nomeclature list:

Nomenclature

Roman symbols

C_p	Constant-pressure specific heat
Le	Lewis number
Pr	Prandtl number
Sc	Schmidt number

Greek symbols

α	Thermal diffusivity
$\mathcal{D}$	Mass diffusivity
λ	Thermal conductivity
μ	Dynamic viscosity
ν	Momentum diffusivity
ρ	Density

13.4 Fun with floats

13.4.1 The subfigure package

Using this package it is possible to include several small figures and tables within a single figure or table environment. This provides a convenient way of refering the subfigures; adding entries to the table of figures is also made possible.

13.4.1.1 Usage

The package can be loaded by using

```
\usepackage[⟨options⟩]{subfigure}
```

in the document preamble. The various options included in the package are:

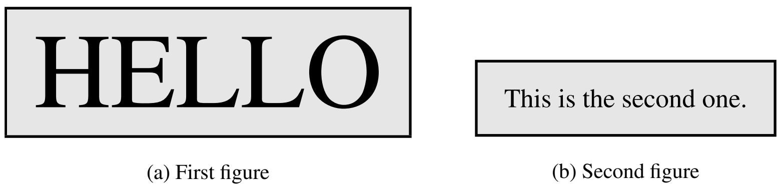
normal	Provides ‘normal’ captions; this is the default.
hang	Gives a hanging indentation to the caption paragraph.

center	This causes each line of the caption paragraph to be separately centered.
centerlast	Only the last line of the caption paragraph is centered.
nooneline	A caption line fitting on a single line is centered by default; this option causes the same to be left-justified.
scriptsize, . . . , Large	Sets the font size of the captions.
up, it, sl, sc, md, bf, rm, sf, tt	Sets the font attributes of the captions.

The following commands can be used within a `figure` or `table` environment to create subfigures or subtables. The amount of vertical space between the figure and the caption can be controlled by `\subfigcapskip`. By default, this is set to `10pt`. `\subfigbottomskip` denotes the amount of vertical space added at the bottom; the default value is `10pt`.

13.4.1.2 Examples

The following example makes use of the `subfigure` package to put two figures side by side.



(a) First figure (b) Second figure

Figure 13.2 A simple example

Note that the subfigures 13.2(a) and 13.2(b) in the figure 13.2 are aligned along the bottom. These are obtained using the following code:

```
\begin{figure}
\centering
\subfigure[First figure]{\label{fig-a}...}\hspace{.75cm}
\subfigure[Second figure]{\label{fig-b}...}
\caption{A simple example}\label{two-figs}
\end{figure}
```

It is similarly possible to obtain tables side by side.

One	Two
Three	Four

(A) Table 1

Another	small	table
But	slightly	bigger
than	previous	one

(B) Second table

Table 13.1 This is it!

13.4.2 Rotating figures

The `rotating` package provides the `\rotcaption` command which makes it possible to rotate the caption thus enabling to typeset a figure in landscape mode.

```
\begin{figure}
\centering
\begin{minipage}[c]{0.6in}
\rotatebox{90}{\fcolorbox{orange}{gray10}{\myfont TEST}}
\end{minipage}
\begin{minipage}[c]{0.4in}
\rotcaption{A rotated figure.}
\end{minipage}
\end{figure}
```

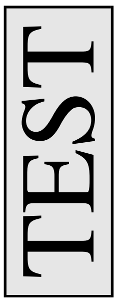


Figure 13.3: A rotated figure.

Another option to obtain the rotated caption is to use the command `\rotatebox` in the same way as in the previous example and include the argument in a `\parbox`. The `rotating` package also provides two environments `sidewaysfigure` and `sidewaystable` which are very similar to the regular `figure` and `table` environments except that these turn the contents through 90 degrees counterclockwise. The package also provides the `turn` environment that allows to rotate the contents through an arbitrary angle.

13.5 Items and lists

13.5.1 The shortlst package

The `shortlst` package is very useful for typesetting a list of short items. The regular `itemize` environment leaves

- a lot
- of
- white
- space.

The `shortlst` package provides the following environments:

- `shortitemize`
- `shortenumerate`
- `runenumerate`
- `runitemize`

The `shortitemize` and the `shortenumerate` environments can be used for small list items in a manner very similar to the regular `itemize` and `enumerate` environments. The following example illustrates the use of `shortitemize`:

```
\begin{shortitemize}
\item{the {\sf itemize} environment}
\item{leaves}
\item{a lot}
\item{of}
\item{white space.}
\end{shortitemize}
```

- the itemize environment
- leaves
- a lot
- of
- white space.

The environment also provides an optional argument that can be used to specify the width of the default allotment of space (the default is `65pt`). For example, using `\begin{shortitemize}[the {\sf itemize} environment]` will produce:

■ the itemize environment	■ leaves	■ a lot
■ of	■ white space.	

Instead of using the optional argument, the width of the item can also be set using the command `\shortitemwidth`. The use of the `shortenumerate` environment is very similar to that of `shortitemize`. Both these environments can be a part of an item of a regular list environment. However, note that no list environment can be used within any of these list environments. The other two environments, `runenumerate` and `runitemize`, provided with this package can be used for items that do not need a displayed paragraph. The following example illustrates the use of the `runenumerate` environment:

```
You have three choices:
\begin{runenumerate}
\item wash your hands,
\item postpone it until tomorrow, or
\item \label{choice}stay dirty.
\end{runenumerate}
I choose \ref{choice}!
```

You have three choices:(1) wash your hands,(2) postpone it until tomorrow, or (3) stay dirty.I choose 3!

The commands `\parbox` or `\minipage` can be used in case a few lists are too long to fit on a single line. The length `\labelsep` denotes the separation between the label and the item; and `\labelwidth` denotes the width of the labels. `\runitemsep` denotes the space between the items of a `\runenumerate` or `\runitemize` environment.

13.5.2 The multienum package

This package is especially useful for generating an enumerated list involving short items, e.g. the solutions manual for a text. The package provides the `multienumerate` environment that has an optional argument for enumerating even-only or odd-only arrays.

```
\begin{multienumerate}[⟨option⟩] ... \end{multienumerate}
```

where the ⟨*option*⟩ `evenlist` produces an enumerated array using only even numbers, the ⟨*option*⟩ `oddlist` produces one using only odd numbers, and no ⟨*option*⟩ produces a consecutively enumerated array. Each row of the enumerated array is set using commands of the following form:

\mitemx{}	A single item in the row.
\mitemxx{}{}	Two items in the row.
\mitemxxx{}{}{}	Three items in the row.
\mitemxox{}{}	Three items in the row with the center item space left blank so the first item can extend into its space.
\mitemxxo{}{}	Three items in the row with the last item left blank so the second item can extend into its space.
\mitemxxxx{}{}{}{}	Four items in the row.
\mitemxoxx{}{}{}	Four items in the row with the second space left blank so the first item can extend into its space.
\mitemxxox{}{}{}	Four items in the row with the third space left blank so the second item can extend into its space.
\mitemxxxo{}{}{}	Four items in the row with the last space left blank so the third item can extend into its space.

There can be a maximum of 4 enumerated entries in a single line[3]. The character `x` in the above commands refer to an entry, while the character `o` refers to a blank entry, and the space for that entry gets utilized by the previous entry.

The following example illustrates the use of the different commands that can be used to generate the enumerated list:

2. 3 X 2
4. 2
6. 3
8. 1
10. Not defined
12. $\begin{pmatrix}-5\\1\\5\end{pmatrix}$
14. $\begin{pmatrix}20\\-5\end{pmatrix}$
16. $\begin{pmatrix}-2\\4\\0\end{pmatrix}$
18. $\begin{pmatrix}41\\52\end{pmatrix}$
20. $\begin{pmatrix}12\\8\\4\end{pmatrix}$
22. $\arccos(9/\sqrt{85}) \approx 0.22$ radians
24. $\sqrt{10}$
26. $\sqrt{3}$
28. Not defined
30. $x = 2$ and $y = 1/2$
32. $C + A = 2\pi r + \pi r^2$
34. $\begin{pmatrix}-1\\2\end{pmatrix}$

The code that produced the above enumerated list is given below[4]:

```
\def\Matrix#1{\begin{pmatrix}#1\end{pmatrix}}
\begin{multienumerate}[evenlist]
\mitemxxxxx{3 X 2}{2}{3}{1}{Not defined}
\mitemxxxxx{$\Matrix{-5 \cr 1 \cr 5}$}{$\Matrix{20 \cr -5}$}%
{$\Matrix{-2 \cr 4 \cr 0}$}{$\Matrix{41 \cr 52}$}{$\Matrix{12 \cr 8 \cr 4}$}
\mitemxoxxx{arccos(9/$\sqrt{85}$) $\approx$ 0.22 radians}%
{$\sqrt{10}$}{$\sqrt{3}$}{Not defined}
\mitemxoxox{$x = 2$ and $y = 1/2$}{$C + A = 2\pi r + \pi r^2$}{$\Matrix{-1 \cr 2}$}
\end{multienumerate}
```

13.6 Some more tricks

13.6.1 The romannum package

The `romannum` package can be used to change the numbers generated by LaTeX for chapters, sections, equations, list items, footnotes, etc. from arabic to roman numerals. The package options, as described below, can be used to typeset uppercase or lowercase roman numerals.

Section	Sectional numbers in uppercase roman.
section	Sectional numbers in lowercase roman.
Equation	Equation numbers in uppercase roman.
equation	Equation numbers in lowercase roman.
Caption	Table and Figure caption numbers in uppercase roman.
caption	Table and Figure caption numbers in lowercase roman.
Footnote	Footnote numbers in uppercase roman.
footnote	Footnote numbers in lowercase roman.
Enumerate	First level items in uppercase roman and third level items in lowercase roman.

[3] The example below illustrates 5 enumerated entries in a line; this is obtained by adding some simple macros in the package.

[4] The `\mitemxoxxx` and `\mitemxoxox` commands have been defined in a similar manner to the other commands in the package.

enumerate	First level items in lowercase roman and third level items in uppercase roman.
Year	The year number from the `\today` command in uppercase roman.
Day	The year number from the `\today` command in uppercase roman and the day number in uppercase roman.
day	The year number from the `\today` command in uppercase roman and the day number in uppercase roman.
Most	A shorthand option equivalent to using all these options: `Section`, `Equation`, `Caption`, `Footnote`, `Enumerate`; that is, all the uppercasing options except for `Year` and `Day`.
most	A shorthand option equivalent to using all these options: `section`, `equation`, `caption`, `footnote`, `enumerate`; that is, all the lowercasing options except for `day`.

13.6.2 The epigraph package

A good question is never answered. It is not a bolt to be tightened into place but a seed to be planted and to bear more seed toward the hope of greening the landscape of idea.

John Ciardi

This package provides fancy styles for typesetting quotes just after a sectional heading. The epigraphs can be typeset either at the left, the center, or the right of the typeblock. The command

```
\epigraph{⟨text⟩}{⟨source⟩}
```

typesets an epigraph using ⟨*text*⟩ as the main text of the epigraph, and the ⟨*source*⟩ as it's reference. The package provides the following commands:

\qitem	The `\qitem{`⟨*text*⟩`}{`⟨*source*⟩`}` command is used in the epigraphs environment in order to specify each epigraph in the list. It's use is essentially similar to the `\item` command in the ordinary list environments.
\epigraphwidth	It denotes the width of the epigraph; the default is `0.4\textwidth`.
\textflush	It controls the ⟨*text*⟩ typesetting style; set to `flushleft` by default.
\epigraphflush	The default position of the epigraphs is at the right hand side of the textblock (set to `flushright`). Using this command, the position of the textblock can be changed.
\sourceflush	It controls the position of the ⟨*source*⟩; default is `flushright`.
\epigraphsize	It can be used to redefine the fontsize in which the epigraphs are typeset; default is `small`.
\epigraphrule	This denotes the thickness of the rule drawn between the ⟨*text*⟩ and the ⟨*source*⟩; default is `0.4pt`.
\beforeepigraphskip, \afterepigraphskip	These commands control the amount of vertical space instered before and after the typeset epigraphs; default value for both the lengths is `0.5\baselineskip`.

14 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.

14.1 Footnotes

Footnotes are generated with the command

```
\footnote{footnote_text}
```

which comes immediately after the word requiring an explanation in a footnote. The next {*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:

> Footnotes in a minipage are numbered using lowercase letters.[a]
> This text references a footnote at the bottom of the page.[3]
>
> [a] Inside minipage

```
\begin{minipage}{5cm}
Footnotes in a minipage are
numbered using lowercase
letters.\footnote{Inside
minipage} \par This text
references a footnote at
the bottom of the
page.\footnotemark \end{minipage}
\footnotetext{At bottom of page}
```

The footnote numbering is incremented throughout the whole 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

14.1.1 Footnotes in Tabular Material

Footnotes appearing inside tabular material are not typeset by standard LaTeX. Only tabularx and longtable environments will treat footnotes correctly. But footnotes used in these tables won't come just following the tables, but appear at the bottom of the page just like the footnotes used in the text. But in longtable you can place the footnotes as table notes by placing the longtable in a minipage. See below:

Table 14.1: PostScript type 1 fonts

Courier[a]	cour,courb,courbi,couri
Nimbus[b]	unmr, unmrs
URW Antiqua[b]	uaqrrc
URW Grotesk[b]	ugqp
Utopia[c]	putb, putbi, putr, putri

[a] Donated by IBM.
[b] Donated by URW GmbH.
[c] Donated by Adobe.

```
\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.} &
unmr, unmrs \\
URW Antiqua\footnotemark[\value{mpfootnote}]
& uaqrrc\\
URW Grotesk\footnotemark[\value{mpfootnote}]
& ugqp\\
Utopia\footnote{Donated by Adobe.} & putb,
putbi, putr, putri
\end{longtable}
\end{minipage}
```

You can also put your tabular or array environment inside a minipage environment, since in that case footnotes are typeset just following that environment. Note the redefinition of \thefootnote that allows us to make use of the \footnotemark command inside the minipage environment. Without this redefinition \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}{\bfseries%
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.} & unmr, 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}
```

PostScript type 1 fonts	
Courier[a]	cour, courb, courbi, couri
Charter[b]	bchb, bchbi, bchr, bchri
Nimbus[c]	unmr, unmrs
URW Antiqua[c]	uaqrrc
URW Grotesk[c]	ugqp
Utopia[d]	putb, putbi, putr, putri

[a] Donated by IBM.
[b] Donated by Bitstream.
[c] Donated by URW GmbH.
[d] 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 minipage width argument might be necessary.

Another way to typeset table notes is with the package 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 your tabular material. In this sense, the `threeparttable` environment is similar to the nonfloating `table` environment.

```
\begin{threeparttable}
\caption{\textbf{PostScript type 1 fonts}}
\begin{tabular}{ll}
Courier\tnote{a} &
cour, courb, courbi, couri\\
Charter\tnote{b} &
bchb, bchbi, bchr, bchri \\
Nimbus\tnote{c} & unmr, unmrs \\
URW Antiqua\tnote{c} & uaqrrc\\
URW Grotesk\tnote{c} & ugqp\\
Utopia\tnote{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**

Courier[a]	cour, courb, courbi, couri
Charter[b]	bchb, bchbi, bchr, bchri
Nimbus[c]	unmr, unmrs
URW Antiqua[c]	uaqrrc
URW Grotesk[c]	ugqp
Utopia[d]	putb, putbi, putr, putri

[a] Donated by IBM.
[b] Donated by Bitstream.
[c] Donated by URW GmbH.
[d] Donated by Adobe.

14.1.2 Customizing footnotes

If the user wishes the footnote numbering to be reset to 1 for reach `\section` command with the article class, this may be achieved with putting

```
\setcounter{footnote}{0}
```

before every sections or using the following command at preamble[4]

```
\@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}{\number_style{footnote}}
```

where *number_style* is one of the counter print commands. \arabic, \roman, \Roman, \alph, or \Alph. However, for the counter `footnote`, there is an additional counter print command available, \fnsymbol, which prints the counter values 1–9 as one of nine symbols:

$\star$ † ‡ § ¶ ‖ $\star\star$ †† ‡‡

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, here which allows up to 12 footnotes without resetting the counter

[4] This command will only work within \makeatletter and \makeatother

```
\makeatletter
\def\@fnsymbol#1{\ensuremath{\ifcase#1\or *\or \dagger\or \ddagger\or
   \mathsection\or \mathparagraph\or \|\or **\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 *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[**],

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
For example\footnote[7]{The 7th symbol .... 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.

14.1.3 Footnote style parameters

The appearance of the standard footnote can be changed by customizing the parameters shown below:

\footnotesize	The font size used inside footnotes
\footnotesep	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}}
```

[**] The 7th symbol appears as the footnote marker.

14.2 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

This
is a
margi-
nal
note

```
 ... command occurs\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, then it is used for the left margin, while the second (mandatory) argument is used for the right.

There are few important things to understand when using marginal notes. Firstly, `\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

14.2.1 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 LaTeX. This query is produced with the following command.

Hey!
Look

```
For example \query{\LaTeX}{Hey!\\ Look}{}. This ...
```

14.2.2 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

14.3 Endnotes

Scholarly works usually group notes at the end of each chapter or at the end of the documents. These are called endnotes. Endnotes are not supported in standard LaTeX, 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.[1] This is simple text.[2] This is simple text.[3]

Notes

[1]The first endnote.
[2]The second endnote.
[3]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.}

\theendnotes\bigskip
This is some more simple text
```

15 Bibliographic Databases

Bibliographic Database is a database where 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.

15.1 The BibTeX program

BibTeX is an auxiliary program to LaTeX that automatically constructs a bibliography for LaTeX document form 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.

15.2 BibTeX Style files

plain	Standard BibTeX style. Entries sorted alphabetically with numeric labels.
unsrt	Standard BibTeX style. Similar to plain, but entries are printed in order of citation, rather than sorted. Numeric labels are used.
alpha	Standard BibTeX style. Similar to plain, but the labels of the entries are formed from the author's name and the year of publication.
abbrv	Standard BibTeX style. Similar to plain, but entries are more compact, since first names, month, and journal names are abbreviated.
acm	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.
apalike	Alternative BibTeX style, used by the journals of the American Psychology Association. It should be used together with the LaTeX 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,`	`jtb.bst, kluwer.bst, named.bst,`
`agsm.bst, alpha.bst, amsalpha.bst,`	`named.sty, natbib.sty, natbib.bst,`
`authordate`*i*`.bst, authordate1-4.sty,`	`nature.sty, nature.bst, phcpc.bst,`
`bbs.bst, cbe.bst, cell.bst,`	`phiaea.bst, phjcp.bst, phrmp.bst`
`dcu.bst, harvard.sty, ieeetr.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` file, or else generate our own using the `makebst` program.

15.2.1 Steps for running BibTeX with LaTeX

(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 ifact inputs the .bbl file.

15.3 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:

```
@BOOK{knuth:86a,
 AUTHOR          ="Donald E. Knuth",
 TITLE           ={The \TeX{}book},
 EDITION         ="third"
 PUBLISHER       ="Addison-Wesley",
 ADDRESS         ={Reading, MA},
 YEAR            =1986 }
```

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* shows 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 occur 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 comment or an abstract of the paper. Ignored fields might also be ones that are used by other database programs.

The general syntax for entries in the bibliographic database reads

```
@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

```
@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

`@incollection` 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

15.3.1 Example of a LaTeX file (sample.tex) using bibliographical database (bsample.bib)

```
\documentclass{article}
\pagestyle{empty}
\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~\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}.
An example of multiple
citations~\cite{Eijkhout:1991,Roth:postscript}.
```

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

15.3.2 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
```

16 Table of Contents, Index and Glossary

16.1 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, are all 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 don't 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.

16.1.1 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`.
type	The type of the entry. For the `toc` file the *type* is normally the same as the heading according to whose format an entry must be typeset. For the `lof` or `lot` files, `figure` or `table` is specified.
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}}
```

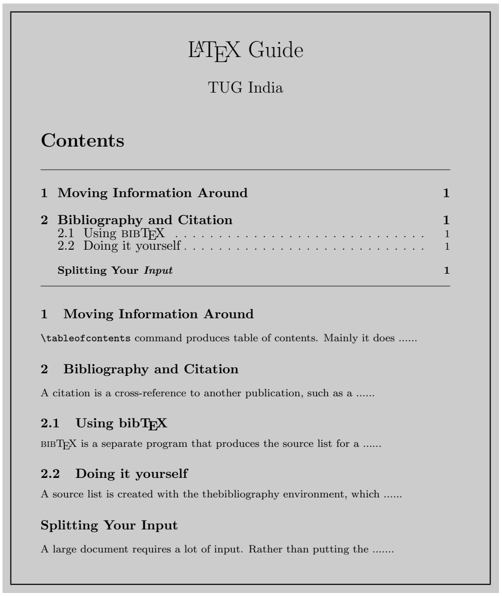
The `\addcontentsline` instruction is usually invoked *automatically* by the document sectioning commands, or by the `\caption` commands. If the entry contains numbered text, then `\numberline` must be used to separate the section number (*number*) from the rest of the text for the entry (*heading*)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

```
\protect\numberline{number}heading
```

```
\documentclass{article}

\def\bibTeX{\textsc{bib}\TeX}

\begin{document}

\title{\LaTeX{} Guide}
\author{TUG India}
\date{}

\maketitle

\tableofcontents

\addtocontents{toc}{\protect\rule{\textwidth}{.2pt}\par}
\section{Moving Information Around}
\verb+\tableofcontents+ command produces table of contents.......

\section{Bibliography and Citation}
A citation is a cross-reference to another publication, such......

\subsection{Using \bibTeX}
\bibTeX\ is a separate program that produces the source list ......

\subsection{Doing it yourself}
A source list is created with the thebibliography ......

\addcontentsline{toc}{section}{\numberline{}Splitting Your \emph{Input}}
\section*{Splitting Your Input}
\addtocontents{toc}{\noindent\protect\rule{\textwidth}{.2pt}\par}

A large document requires a lot of input. Rather than .......

\end{document}
```

Figure 16.1: Input file contains `\tableofcontents` command.

```
\rule {\textwidth }{.2pt}\par
\contentsline {section}{\numberline
      {1}Moving Information Around}{1}
\contentsline {section}{\numberline
      {2}Bibliography and Citation}{1}
\contentsline {subsection}{\numberline
      {2.1}Using \textsc {bib}TeX}{1}
\contentsline {subsection}{\numberline
      {2.2}Doing it yourself}{1}
\contentsline {section}{\hbox
           to\@tempdima {\hfil }
      Splitting Your \emph {Input}}{1}
\noindent \rule {\textwidth }{.2pt}\par
```



LaTeX Guide

TUG India

Contents

1 Moving Information Around 1

2 Bibliography and Citation 1
2.1 Using BIBTeX 1
2.2 Doing it yourself 1

Splitting Your *Input* 1

1 Moving Information Around

`\tableofcontents` command produces table of contents. Mainly it does ......

2 Bibliography and Citation

A citation is a cross-reference to another publication, such as a ......

2.1 Using bibTeX

BIBTeX is a separate program that produces the source list for a ......

2.2 Doing it yourself

A source list is created with the thebibliography environment, which ......

Splitting Your Input

A large document requires a lot of input. Rather than putting the ....

Figure 16.2: Output `.toc` file and `.dvi`

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

```
\addcontentsline{lof}{figure}{\protect\numberline{\thefigure}captioned text}
```

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:

```
\chapter*{Forward}
\addcontentsline{toc}{chapter}{\numberline{}Forward}
```

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.

16.1.2 Typesetting a Contents List

As discussed above, contents lists consists 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:

```
\contentsline{type}{text}{page}
```

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	pagenumber.

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:

```
\contentsline {section}
  {\numberline {2.4}Structure of the Table of Contents}{31}
```

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

```
\@dottedtocline{level}{indent}{numwidth}{text}{page}
```

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.

16.1.3 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 16.1: Summary of the `minitoc` parameters

\dominitoc	must be put just in front of \tableofcontents, to initialize the minitoc system (Mandatory).
\faketableofcontents	this command replaces \tableofcontents when you want mini-tocs but not table of contents.
\minitoc	this command must be put right after each \chapter command where a minitoc is desired.
minitocdepth	a LaTeX counter that indicates how many levels of headings will be displayed in the minitoc (default value is 2).
\mtcindent	the length of the left/right indentation of the minitoc (default value is 24pt).
\mtcfont	command defining the font that is used for the minitoc entries (The default definition is a small roman font).

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.

16.2 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 `\index` command causes LaTeX to write an entry in the `.idx` file. This command writes the text given as an argument, in the `.idx` file. This `.idx` will be generated only if we give `\makeindex` command in the preamble otherwise it will produce nothing.

```
\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 `\index` command.

(2) Include the makeidx package with an `\usepackage` command and put `\makeindex` commad at the preamble.

(3) Put a `\printindex` command where the index is to appear, normally before `\end{document}` command.

(4) LaTeX *file*. Then a raw index (`file.idx`) will be generated.

(5) Then Run `makeindex`. (makeindex *file.idx* or makeindex *file*). Then two more files will be generated, *file.ind* which contains the index entries and *file.ilg*, a transcript file.

(6) Then again run LaTeX. Now you can see in the dvi that the Index has been generated in a new page.

```
Page vi:   \index{animal}
Page 5:    \index{animal}
Page 6:    \index{animal}
Page 7:    \index{animal}
Page 11:   \index{animalism|see{animal}}
Page 17:   \index{animal@\emph{animal}}
           \index{mammal|textbf}
Page 26:   \index{animal!mammal!cat}
Page 32:   \index{animal!insect}
```

(a) The input file

```
\indexentry{animal}{vi}
\indexentry{animal}{5}
\indexentry{animal}{6}
\indexentry{animal}{7}
\indexentry{animalism|seeanimal}{11}
\indexentry{animal@\emph{animal}}{17}
\indexentry{mammal|textbf}{17}
\indexentry{animal!mammal!cat}{26}
\indexentry{animal!insect}{32}
```

(b) The `.idx` file

```
\begin{theindex}
  \item animal, vi, 5–7
    \subitem insect, 32
    \subitem mammal
      \subsubitem cat, 26
  \item \emph{animal}, 17
  \item animalism, \see{animal}{11}
\indexspace
  \item mammal, \textbf{17}
\end{theindex}
```

(c) The `.ind` file

```
animal, vi 5–7
    insect, 32
    mammal
      cat, 26
animal, 17
animalism, see animal

mammal, 17
```

(d) The typeset output

Figure 16.3: Stepwise development of index processing

16.2.1 Simple Index Entries

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

```
\index{index_entry}
```

16.2.2 Sub Entries

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

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

This will come as

```
dimensions
    rule
        width, 5
```

16.2.3 Page Ranges and Cross-References

You can specify a page range by putting the command `\index{...|(}` at the beginning of the range and `\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 `see` encapsulator. Since "see" entry does not print any page number, the commands `\index{...|see{...}}` can be placed anywhere in the input file after the `\begin{document}` command. For practical reasons, it is convenient to group all such cross-referencing commands in one place.

```
fonts                              Page ii:   \index{table|(}
    Computer Modern, 13–25         Page xi:   \index{table|)}
    math, see math, fonts          Page 5:    \index{fonts!PostScript|(}
    PostScript, 5                             \index{fonts!PostScript|)}
table, ii–xi, 14                   Page 13    \index{fonts!Computer Modern |(}
                                   Page 14:   \index{table}
                                   Page 17:   \index{fonts!math|see{math, fonts}}
                                   Page 21:   \index{fonts!Computer Modern}
                                   Page 25:   \index{fonts!Computer Modern|)}
```

Figure 16.4: Page range and cross-referencing

16.2.4 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: *key@visual*, where *key* determines the alphabetical position and the string *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 *n* after a page number denoting that the item appears in a footnote on that page. *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 \index{keyword|xxx} will produce a page number of the form \xxx{n}, where *n* is the page number in question. Similarly, the command \index{keyword|(xxx)} will generate a page range of the form \xxx{n-m}

```
\newcommand{\nn}[1]{#1n}
```

delta, 14	Page ii:	\index{tabular|textbf}
δ, 23	Page 5:	\index{ninety-five}
delta wing, 16	Page 7:	\index{tabbing}
flower, 19	Page 14:	\index{delta}
ninety, 26	Page 16:	\index{delta wing}
xc, 28	Page 19:	\index{flower@\textbf{flower}}
ninety-five, 5	Page 21:	\index{tabular|textit}
tabbing, *7*, *34–37*	Page 22:	\index{tabular|nn}
tabular, **ii**, *21*, 22n	Page 23:	\index{delta@δ}
`tabular` environment, 23		\index{tabular@\texttt{tabular} environment}
	Page 26:	\index{ninety}
	Page 28:	\index{ninety@xc}
	Page 34:	\index{tabbing|(textit}
	Page 36:	\index{tabbing|)}

Figure 16.5: Controlling the presentation form

16.2.5 Printing those Special Characters

To typeset one of the characters having a special meaning to *MakeIndex* (!, ”, @, 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 !, @, ”, or | characters are treated like ordinary characters, losing their special meaning. The ” preceding a quoted character is deleted before the entries are alphabetised.

@ sign, 2		\index{bar@\texttt{”|}|see{vertical bar}}
|, *see* vertical bar	Page 1:	\index{quote (\verb+”””+)}
exclamation (!), 4		\index{quote@\texttt{”””} sign}
Ah!, 5	Page 2:	\index{atsign@\texttt{”@} sign}
Mädchen, 3	Page 3:	\index{maedchen@M\”{a}dchen}
quote (”), 1	Page 4:	\index{exclamation (”!)}
” sign, 1	Page 5:	\index{exclamation (”!)!Ah”!}

16.6 Printing those special characters

16.3 Glossary

A ‘glossary’ is a special index of terms and phrases alphabetically ordered together with their explanations. To help set up a glossary, LaTeX offers the commands

`\makeglossary`	in the preamble and
`\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{`*glossary-entry*`}{`*pagenumber*`}`

The information 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.

17 Typesetting Theorems

17.1 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 `\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

Theorem 1. *The sum of the angles of a triangle is* 180°.

you first specify

```
\newtheorem{thm}{Theorem}
```

and then type

```
\begin{thm}
The sum of the angles of a triangle is $180^\circ$.
\end{thm}
```

Note that in the command `\newtheorem` the first argument can be any name you fancy, instead of the `thm` given here. Also, it is a good idea to keep all your `\newtheorem` commands together in the preamble.

The `\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 1.1 (the first theorem of the first section) rather than a plain 1, then you must specify

```
\newtheorem{thm}{Theorem}[section]
```

in the `\newtheorem` command. Then the same input as above for the theorem produces

Theorem 17.1.1. *The sum of the angles of a triangle is* 180°.

The next **Theorem** will be numbered 1.2, the third **Theorem** in the fourth section will be numbered 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 17.1.2. *The sum of the angles of a triangle is* 180°.

An immediate consequence of the result is the following

Corollary 17.1.3. *The sum of the angles of a quadrilateral is* 360°.

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
\begin{cor}
The sum of the angles of a quadrilateral  is $360^\circ$.
\end{cor}
```

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 17.1.4 (Euclid). *The sum of the angles of a triangle is* 180°.

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 17.1.5. [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 17.1.6 (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

THEOREM 17.1.1 (EUCLID). *The sum of the angles of a triangle is* 180°.

Such customization is necessitated not only by the aesthetics of the author but often by the whims of the designers in publishing houses also.

17.2 Designer Theorems— The amsthm package

The package amsthm affords a high level of customization in formatting theorem-like statements. Let's first look at the predefined *s2tyles* available in this package.

17.2.1 Ready made styles

The default style (this is what you get if you don't say anything about the style) is termed `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

```
\usepackage{amsthm}
\newtheorem{thm}{Theorem}[section]
\theoremstyle{definition}
\newtheorem{dfn}{Definition}[section]
\theoremstyle{remark}
\newtheorem{note}{Note}[section]
\theoremstyle{plain}
\newtheorem{lem}[thm]{Lemma}
```

and then type somewhere in your document

```
\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}
```

then you get

Definition 17.2.1. A triangle is the figure formed by joining each pair of three non collinear points by line segments.

Note 17.2.1. A triangle has three angles.

Theorem 17.2.1. *The sum of the angles of a triangle is* 180°.

Lemma 17.2.2. *The sum of any two sides of a triangle is greater than or equal to the third.*

Note how the `\theoremstyle` command is used to switch between various styles, especially the last `\theoremstyle{plain}` command. Without it, the previous `\theoremstyle{remark}` will still be in force when `lem` is defined and so "Lemma" will be typeset in the `remark` style.

17.2.2 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 *environment* which is to be used later. Thus in the example above `remark` is the name of a new style for typesetting theorem like statements and `note` is the name of the environment subsequently defined to have this style. (and `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 *declaration* such as `\scshape` or `\bfseries` and *not* as a *command* such as `\textsc` or `\textbf`. If this is left empty, then the main text font of the document is used.

The next four parameters refer to the *theoremhead*—the part of the theorem like statement consisting of the name, number and the optional note. The fifth parameter *indent* specifies the indentation of *theoremhead* from the left margin. If this is empty, then there is no indentation of the *theoremhead* from the left margin. The next parameter specifies the font to be used for the *theoremhead*. 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 don't 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's now look at a few examples using the eight parameters we've already discussed.

It's almost obvious now how the last theorem in Section 1 (see Page 3) 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{mythm}{Theorem}[section]

\begin{mynewthm}[Euclid]
The sum of the angles of a triangle is $180^\circ$.
\end{mynewthm}
```

This produces

Theorem 17.2.1 (Euclid):
The sum of the angles of a triangle is 180°.

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* γ *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 is for setting this parameter is

```
{commands#1commands#2commands#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's 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

```
\newtheoremstyle{diffnotenonum}{}{}{\itshape}{}%
  {\bfseries}{.}{ }{#1 (\mdseries #3)}

\theoremstyle{diffnotenonum}
\newtheorem{Cauchy}{Cauchy's Theorem}

\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

```
\theoremstyle{diffnotenonum}
\newtheorem{Riemann}{Riemann Mapping THeorem}

\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'll 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's happened? In the `\theoremstyle{diffnotenonum}`, the parameter controlling the *note* part of the *theoremhead* was defined as `(\mdseries #3)` and in the `\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 `\thmname`, `\thmnumber` and `\thmnote` *within* the {⟨*custom-head-spec*⟩} as

```
{\thmname{⟨commands#1⟩}%
    \thmnumber{⟨commands#2⟩}%
     \thmnote{⟨commands#3⟩}
```

Each of these three commands will typeset its argument *if and only if the corresponding argument in the* `\thmhead` *is non empty*. Thus the correct way to get the **Riemann Mapping theorem** in Page 7 is to input

```
\newtheoremstyle{newdiffnotenonum}{}{}%
  {\itshape}{}{\bfseries}{.}{ }%
    {\thmname{#1}\thmnote{ (\mdseries #3)}}

\theoremstyle{newdiffnotenonum}
\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 6 by typing

```
\theoremstyle{newdiffnotenonum}
\newtheorem{newCuchy}{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 6. Now suppose you want to highlight certain theorems from other sources in your document, such as

Axiom 1 in (1). *Things that are equal to the same thing are equal to one another.*

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`.

17.2.3 There's more!

There are some more predefined features in amsthm package. In all the different examples we've seen so far, the *theorem number* comes after the *theorem name*. Some prefer to have it the other way round as in

17.2.1. Theorem (Euclid). *The sum of the angles in a triangle is* 180°

This effect is produced by the command `\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 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

```
\newtheorem*{numlessthm}{Theorem}[section]

\begin{numlessthm}[Euclid]
The sum of the angles in a triangle is $180^$.
\end{numlessthm}
```

to produce

Euclid. *The sum of the angles in a triangle is* 180°.

Note that this could also be done by leaving out `#2` in the *custom-head-spec* parameter of `\newtheoremstyle`, as seen earlier.

We've 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
$n=p_1p_2\dotsm p_k+1$. 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 $p_1,p_2,\dotsc p_k$ so that in either case,
we get a prime different from $p_1,p_2\dotsc p_k$. Thus no finite
set of primes includes all the primes.
\end{proof}
```

to produce the following output

Theorem 17.2.3. *The number of primes is infinite*

Proof. Let $\{p_1, p_2, \dots p_k\}$ 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 $p_1, p_2, \dots p_k$ so that in either case, we get a prime different from $p_1, p_2 \dots p_k$. Thus no finite set of primes can include all the primes. □

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 $\{p_1,p_2,\dotsc p_k\}$ be a finite set of primes. Define
$n=p_1p_2\dotsm p_k+1$. 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 $p_1,p_2,\dotsc p_k$ so that in either case, we
get a prime different from $p_1,p_2\dotsc p_k$. Thus no finite
set of primes includes all the primes.
\end{proof}
```

produces the following

Proof (Euclid): Let $\{p_1, p_2, \dots p_k\}$ 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 $p_1, p_2, \dots p_k$ so that in either case, we get a prime different from $p_1, p_2 \dots p_k$. 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 `\qedsymbol`. If you wish to change it, use `\renewcommand` to redefine the `\qedsymbol`. Thus if you like the original "Halmos symbol" ▮ to mark the ends of your proofs, include

```
\newcommand{\halmos}{\rule{1mm}{2.5mm}}
\renewcommand{\qedsymbol}{\halmos}
```

in the preamble to your document.

Again, the placement of the `\qedsymbol` at the *end* of the last line of the proof is done via the command `\qed`. The default placement may not be very pleasing in some cases as in

Theorem 17.2.4. *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'd be better if this is typeset as

Theorem 17.2.5. *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

$$(x + y)^2 = x^2 + y^2 + 2xy \qquad \square$$

which is achieved by the input shown below:

```
\begin{proof}
This follows easily from the equation
\begin{equation}
  (x+y)^2=x^2+y^2+2xy\tag*{\qed}
\end{equation}
\renewcommand{\qed}{} \end{proof}
```

For this trick to work, you must have loaded the package `amsmath` *without* the `leqno` option. Or, if you prefer

Proof. This follows easily from the equation

$$(x + y)^2 = x^2 + y^2 + 2xy \quad \square$$

Then you can use

```
\begin{proof}
This follows easily from the equation
\begin{equation*}
  (x+y)^2=x^2+y^2+2xy\qed
\end{equation*}
\renewcommand{\qed}{}
\end{proof}
```

17.3 Housekeeping

It’s better to keep all `\newtheoremstyle` commands in the preamble than scattering them all over the document. Then you can divide your `\newtheorem` commands into groups and preface each group with the appropriate `\theoremstyle`.

Moreover, you can keep all your `\newtheoremstyle` in a `.thm` file, for instance `mystyles.thm`, and load it on demand in various documents using

```
\usepackage[mystyles]{amsthm}
```

This method fails if the `amsthm` has already been loaded by the documentclass such as `amsart`. In that case, you will have to use

```
\PassOptionsToPackage{mystyles}{amsthm}
```

Have fun!

References

[1] Euclid, *The Elements*, Greece 300 BC