An Introduction to LaTeX

Bob Dowling[*]

July 24, 1996

[*]University of Cambridge Computing Service, `rjd4@cam.ac.uk`

Contents

List of Tables

1 Introduction

This course will introduce you to running LaTeX on a Unix system. The information about running the `latex` command will be system-specific, but the information about what to put in a LaTeX file (the majority of the course) will be generally applicable. LaTeX is available on a very wide variety of platforms, including Unix machines, VMS machines, IBM-PCs running MS-DOS, MS Windows, or OS/2, and Apple Macintoshes.

LaTeX is a document *description* language. It is sometimes called an "intensional" or "logical" markup language. It is designed to let you specify the *structure* of your document, rather than its appearance. The rules translating structure into appearance are called the "style" and are dealt with separately.

LaTeX is *not* a WYSIWYG ("what you see is what you get") text processor[1]. Rather, it is a batch system where a plain text file with LaTeX instructions in it is processed by a command (almost invariably called "`latex`") to produce an output file which can then be previewed or printed.

The course is only designed to get you *started*. Anything but the most simple use of LaTeX will require you to have easy access to the book written for the program by its author:

Title: "LaTeX—User's Guide and Reference Manual", Second edition

Author: Leslie Lamport

Published: Addison-Wesley, 1994

ISBN: 0-201-52983-1

LaTeX is easy to start to use, but hard to master. In particular, writing your own styles is very hard. See "The LaTeX Companion" for details:

Title: "The LaTeX Companion"

Authors: M. Goossens, F. Mittelbach, A. Samarin

Published: Addison-Wesley, 1993

[1] LaTeX is sometimes called YAFIYGI ("you asked for it; you got it").

ISBN: 0-201-54199-8

The only prerequisite for this course is knowing how to use a text editor on your system of choice.

This course will explain the basics of the LaTeX source for:

- Plain text
- Lists
- Tables
- Diagrams
- Mathematics

A note on versions: There are two versions of LaTeX currently in existence. These are LaTeX2.09 and LaTeX2ε. The latter is the current version and the version this course will cover. If you have the older version, gripe at your system administrator. If you have a large amount of work already done in the older version, don't worry. The newer version spots that the document is old-style and has a fairly good attempt at running in compatability mode. However, all documents you now produce should be in the new form.

We will start with a very simple example. Let us suppose we were going to typeset the book "Treasure Island". The opening paragraph of this book reads as follows:

```
   Squire Trelawney, Dr. Livesey, and the rest
of these gentlemen having asked me to write
down the whole particulars about Treasure
Island, from the beginning to the end,
keeping nothing back but the bearings of the
island, and that only because there is
treasure not yet lifted, I take up my pen in
the year of grace 17-, and go back to the
time when my father kept the "Admiral
Benbow" inn, and the brown old seaman, with
the sabre-cut, first took up his lodging
under our roof.
```

We add to this file two lines at the start and one at the end:

```
\documentclass{book}
\begin{document}
Squire Trelawney, Dr. Livesey,
...
lodging under our roof.
\end{document}
```

The three extra lines are LaTeX commands. These three all begin with a backslash and have an argument in curly brackets (often called "braces"). This is fairly typical for a LaTeX command.

We shall put this in a file `treasure.tex`. Now we need to process this file with the local LaTeX implementation. Here is an example from a Unix system:

```
$ latex treasure
This is TeX, Version 3.1415 (C version 6.1)
(treasure.tex
LaTeX2e <1994/06/01>
(/opt/TeX/lib/texmf/tex/latex2e/base/book.cls
Document Class: book 1994/06/02 v1.2s Standard LaTeX
document class
(/opt/TeX/lib/texmf/tex/latex2e/base/bk10.clo))
No file treasure.aux.
[1] (treasure.aux) )
Output written on treasure.dvi (1 page, 848 bytes).
Transcript written on treasure.log.
$
```

Three files are created by this run of the `latex` command:

`treasure.aux` is a file that can carry certain auxiliary information about the file being processed. In this simple case it contains a single line telling the system to take no special action. Examples of its more powerful use will be given in section 8.

`treasure.log` is a logging file with details of what LaTeX and the underlying TeX program did to process the file. This will not be referred to in the course; the important output will be printed to the screen.

`treasure.dvi` contains the true “output”. This file contains a typeset document in “DeVice Independent” format.

This DVI file can then be previewed directly on an X-terminal by a program called `xdvi`; the command “`xdvi treasure`” will preview the document on the display. Alternatively, the DVI file can be converted into a device-dependent format. The most commonly desired format is PostScript. The command `dvips` converts a DVI file into a PostScript file. The command “`dvips treasure`” creates a file called `treasure.ps` containing the PostScript. This can then be printed to a PostScript printer with the `lpr` command, or previewed on the screen with a suitable PostScript interpreter like `ghostview`. (Syntax: “`ghostview treasure.ps`”.) The output from the `treasure.tex` example is shown below:

> Squire Trelawney, Dr. Livesey, and the rest of these gentlemen having asked me to write down the whole particulars about Treasure Island, from the beginning to the end, keeping nothing back but the bearings of the island, and that only because there is treasure not yet lifted, I take up my pen in the year of grace 17-, and go back to the time when my father kept the ”Admiral Benbow” inn, and the brown old seaman, with the sabre-cut, first took up his lodging under our roof.

There are three problems with the output:

1. “Dr. Livesey” has *sentence* spacing.
2. The dash in “17-” is too short.
3. The leading quotation mark in ”Admiral Benbow” is wrong.

Each of these errors in the output has a simple fix in the input and each will be described in detail later:

1. `Dr.␣Livesey` becomes `Dr.~Livesey`.
2. `17-` becomes `17---`.
3. `"Admiral␣Benbow"` becomes ``` ``Admiral␣Benbow'' ```.

The new output is then as shown below:

> Squire Trelawney, Dr. Livesey, and the rest of these gentlemen having asked me to write down the whole particulars about Treasure Island, from the beginning to the end, keeping nothing back but the bearings of the island, and that only because there is treasure not yet lifted, I take up my pen in the year of grace 17—, and go back to the time when my father kept the "Admiral Benbow" inn, and the brown old seaman, with the sabre-cut, first took up his lodging under our roof.

The book is divided into chapters of which this is the first. If we wanted to do this in the document, we would add the following LaTeX command, just before the main text:

```
\chapter{The Old Sea-Dog at the
``Admiral Benbow''}
```
Squire Trelawney, Dr. Livesey...

The output of this is then:

> **1: The Old Sea-Dog at the "Admiral Benbow"**
>
> Squire Trelawney, Dr. Livesey, and the rest of these gentlemen having asked me to write down the whole particulars about Treasure Island, from the beginning to the end, keeping nothing back but the bearings of the island, and that only because there is treasure not yet lifted, I take up my pen in the year of grace 17—, and go back to the time when my father kept the "Admiral Benbow" inn, and the brown old seaman, with the sabre-cut, first took up his lodging under our roof.

Notes:

- We did not have to explicitly number the chapter. LaTeX takes care of all the drudgery like that.
- You may not like the style of the chapter heading, but for now you have no control over it; that is the job of the style.

2 Error handling

It is fairly likely that you will make errors in the LaTeX document. LaTeX will stop as soon as it notices an error condition and report the fact.

The most common error message is "`Undefined control sequence`". This means that LaTeX does not understand one of the commands typed in. For example consider a LaTeX input document that contained the incorrect command "`\latex`":

```
$ more wrong.tex
\documentclass{article}
\begin{document}
I like \latex.  If only it liked me.
\end{document}

$ latex wrong.tex
This is TeX, Version 3.1415 (C version 6.1)
(wrong.tex
LaTeX2e <1994/06/01>
(/opt/TeX/lib/texmf/tex/latex2e/base/article.cls
Document Class: article 1994/06/02 v1.2s Standard LaTeX
document class
(/opt/TeX/lib/texmf/tex/latex2e/base/size10.clo))
No file wrong.aux.
! Undefined control sequence.
l.3 I like \latex
                 .  If only it liked me.
?
```

The things to note about the output are as follows:

`! Undefined control sequence.`: The line starting with the exclamation mark is the actual error message itself.

`l.3`: LaTeX specifies the line of the source file on which it has observed the error.

`I like \latex`: LaTeX breaks the relevant line just after the error has been observed.

`?` : The query is LaTeX's prompt for input.

There are four common responses to a LaTeX error:

`x`: exit

Exiting leaves LaTeX immediately and gives a report on how many pages of output have already been generated.

`q`: **q**uiet

This causes LaTeX to run through the remaining document, neither reporting nor stopping for any errors. A full transcript is still placed in the `.log` file. "**q**" does *not* mean "quit"!

`h`: **h**elp

Help gives a few lines of help message. It rarely adds anything to what you know and repeated use just infuriates.

`e`: **e**dit

This is the most useful of the options. If your environment is correctly set up in advance you are dropped into the editor of your choice acting on the source file and you are placed at the start of the line where the error was observed.

To use the editing facility, you need to have the `TEXEDIT` environment variable defined as a string that will be a command to start an editor once `%d` has been replaced by the line number and `%s` has been replaced with the file name. Some examples for the most common editors are:

- `vi +%d %s`
- `emacs +%d %s`
- `pico +%d %s`
- `ee %s -opt m%d`

The syntax for setting `TEXEDIT` depends on the shell:

```
/bin/sh: TEXEDIT='vi +%d %s'; export TEXEDIT

/bin/ksh: TEXEDIT='vi +%d %s'; export TEXEDIT

/bin/bash: export TEXEDIT='vi +%d %s'
```

```
/bin/csh: setenv TEXEDIT 'emacs +%d %s'

/bin/tcsh: setenv TEXEDIT 'emacs +%d %s'
```

LaTeX will also warn of certain cases where it has been unable to typeset the document as well as it would have liked. The two most typical warnings are "`Underful vbox`" and "`Overful hbox`". The former means that LaTeX had to stretch the layout vertically too much to fill the page. The latter means that LaTeX could find no suitable word breaks or hyphenation points and that as a result, one of the lines in the output is sticking out a bit too far.

For more details of error handling in LaTeX consult section 2.3 and chapter 8 of Lamport's LaTeX book.

3 Structure of a LaTeX Document

A LaTeX source file has the format shown here:

```
\documentclass{class}
Preamble
\begin{document}
Body
\end{document}
```

We shall consider these components one at a time:

3.1 Document class

- `\documentclass{`*class*`}`
- `\documentclass[`*option,option,...*`]{`*class*`}`

The first active line is always a `\documentclass` declaration. This determines what LaTeX commands will be available, what the layout will be, etc. The classes available with basic LaTeX are:

- `book`
- `report`
- `article`
- `letter`
- `slides`

The three classes `book`, `report` and `article` are designed for the writing of entire books, reports (basically small books) and articles (entries for journals). This set of course notes is written in article style. The letter class is used for writing letters and the slides class for producing slides. This course only covers the first three and uses article as its example throughout.

The class used may take options. If so then these are specified in square brackets before the name of the class. Some of the more commonly used options are:

`10pt,11pt,12pt` Selects the default type size.

`a4paper,a5paper,b5paper` Selects the paper size. The default is the U.S.A. "letter".

`draft` Causes a thick black line to be drawn to the right of any overlong lines.

`twocolumn` Specifies two columns of text per page.

`titlepage,notitlepage` In article mode the title is placed at the top of the first page by default. If the `titlepage` option is specified, the title gets a page to itself. In article and book style, the title gets a titlepage unless the `notitlepage` option is specified.

A complete list is given in Lamport's LaTeX book in section C.5.1.

3.2 The preamble

The preamble is the section before the main body of the document. This section is the place to configure other options and to specify parameters that will alter the whole document.

N.B. Commands in the preamble must not produce any output.

3.2.1 Layout

The layout of the document should be left to the class to determine but facilities are available for changing certain parameters within the class. If any such changes are made they should be made in the preamble.

3.2.2 Packages

External sets of LaTeX commands called "packages" can be loaded in the preamble. These are loaded with the command

```
\usepackage{package}
```

Some of the more commonly used packages are described below:

`amstex`, `latexsym`: Add some commands and symbols for mathematical work.

Some of these packages take options which are passed with the syntax

```
\usepackage[option,option]{package}
```

A fuller list of packages is given in section C.5.2 of Lamport's LaTeX book.

3.3 The Document

The main body of the document is everything between the `\begin{document}` and `\end{document}` statements. This is the only part of the LaTeX input file that may produce any output.

The body of the document has various possible components:

Plain text: Plain text, with no special characters is treated very simply as we have seen in the introductory example.

Specially interpreted characters: We have already seen `~` used to produce a non-breaking, inter-word space and ` `` ` and `''` to generate the quotation marks.

LaTeX commands: We will see a number of LaTeX commands. An example is `\LaTeX` which generates the silly "LaTeX" logo.

4 Structure of LaTeX Commands

4.1 "Backslash" commands

A LaTeX command that starts with a backslash falls into one of two categories:

Either

- it begins with a backslash (\)
- followed by a *single* non-alphabetic character

or

- it begins with a backslash (\)
- followed by a sequence of alphabetic characters
- terminated by a non-alphabetic character
- any number of trailing spaces are swallowed

Examples:

- `A \% is a percentage character.`
  - "A % is a percentage character."
  - The `%` character is a comment character normally, marking the text upto and including the end of line as a comment.
  - The `\%` command gives the character.
  - The `\` is followed by the `%` which is *non-alphabetic* so LaTeX processes the following space as simple text rather than as a command name.
- `I like \LaTeX.`
  - "I like LaTeX."
  - The `\LaTeX` command generates the logo.
  - The `\` was followed by an *alphabetic* character, "`L`", so the sequence of characters "`LaTeX`" is taken as the command name.

  - The command name was terminated by the first non-alphabetic character, "`.`" which is processed as normal text because it is not a space.

- `But \LaTeX doesn't like me.`
  - "But LaTeXdoesn't like me."
  - *WRONG!*
  - The command name was terminated by the first non-alphabetic character, a space, which terminated the command name but which was *swallowed*.
  - As a result, there is no space between the "LaTeX" and the "doesn't".
  - Multiple spaces won't help.

- `But \LaTeX\ doesn't like me.`
  - "But LaTeX doesn't like me."
  - The command name was terminated by the first non-alphabetic character, "`\`" which is not a space so is not swallowed.
  - The "`\␣`" command generates an inter-word space.
  - A space is not an alphabetic character and so `\␣` is a simple, "single character" command like `\%`.

4.2 Command arguments

Some LaTeX commands require arguments. For example:

- `\documentclass{article}`
- `\frac{12345}{67890}`

  $\frac{12345}{67890}$

Note the use of the braces (`{...}`) for grouping together the characters of an argument. If the fraction example had been "`\frac12345 67890`" the output would have been "$\frac{1}{2}$34567890".

Optional arguments are specified in square brackets. For example:

- `\documentclass[a4paper,12pt]{article}`

5 Characters in LaTeX

5.1 Simple Characters

- `a–z, A–Z`
- `0–9`
- `, ; . : ? ! ‘ ’ ( ) / * @`
- `[ ]`

Table 1: Characters treated normally

All alphabetic characters that are not part of a command name are treated normally. The digits are always treasted normally.

- `[ ]`
- `# $ % & _ { }`
- `^ \ ~`
- `-`

Table 2: Special characters

Most punctuation is treated normally, but the rest needs special treatment.

Square brackets are normally treated as ordinary characters, except when they follow a command that takes optional arguments. If they do, enclose them in curly brackets. For example, under certain circumstances there is a command called “`\item`” which takes an optional argument. So, to follow it with a square bracket one would use “`\item{[}`”. `\item` takes no ordinary arguments and so does not process the contents of the curly brackets as an argument.

The set of characters “`#`”, “`$`”, “`%`”, “`&`”, “`_`”, “`{`” and “`}`” all have special significance in LaTeX, as will be explained later, but can be generated by by the LaTeX commands “`\#`”, “`\$`”, “`\%`”, “`\&`”, “`\_`”, “`\{`” and “`\}`”.

The set of characters “`^`”, “`\`” and “`~`” are not available in simple text mode. See section 7.4 on “verbatim” input for a description on how to get at these characters.

5.2 LATEX commands that generate characters

There are certain character combinations that have special meaning. For example “‘‘”, “’’”, “?‘” and “!‘” give quotation marks, and the Spanish inverted question and exclamation marks.

LATEX is capable of three distinct “dashes”. The simple hyphen is the shortest of the three and is obtained by simply typing a dash: `co-operate` gives “co-operate”. The slightly longer “n-dash” is used for ranges and is given by a double dash in the input. The “m-dash” used for parenthetical comments is given by a triple dash. Consider:

`a-a`	a-a
`a--a`	a–a
`a---a`	a—a

Table 3: The three dashes: hyphen, n-dash and m-dash

Other special characters are generated by other standard LATEX commands. Table 4 contains a list of those that appear in conventional text.

`\dag`	†	`\ddag`	‡
`\S`	§	`\P`	¶
`\pounds`	£	`\ss`	ß
‘‘	“	’’	”
`?‘`	¿	`!‘`	¡
`--`	–	`---`	—
`\aa`	å	`\AA`	Å
`\ae`	æ	`\AE`	Æ
`\l`	ł	`\L`	Ł
`\o`	ø	`\O`	Ø
`\oe`	œ	`\OE`	Œ
`\i`	ı	`\j`	ȷ

Table 4: Commands giving special characters

The “backslash” commands all need a trailing space or other special attention.

```
encyclop\ae␣dia
```

```
encyclop{\ae}dia
```

encyclopædia

5.3 Accented Characters

`\'{o}`	ó	``\`{o}``	ò
`\^{o}`	ô	`\"{o}`	ö
`\~{o}`	õ	`\.{o}`	ȯ
`\={o}`	ō	`\H{o}`	ő
`\u{o}`	ŏ	`\v{o}`	ǒ
`\t{oo}`	o͡o	`\c{o}`	o̧
`\d{o}`	ọ	`\b{o}`	o̲

Table 5: Commands giving accented characters

- Table 5 gives a list of accent instructions.
- Accents can be placed over arbitrary characters.
- Use `\i` and `\j` with accents:

  `\"i` gives i̇̈

  `\"\i` gives ï
- Stacking is not generally possible:

  `\c{\'{o}}` gives ó̧

  `\'{\c{o}}` gives ´o̧

5.4 Horizontal White Space in Output

LaTeX has two principal types of break in its linear text:

word breaks are caused by one or more spaces or by the end of the input line

sentence breaks are caused by a word break after a "!", "?" or "." *unless* the punctuation follows a capital letter (because LaTeX then assumes it is a word break after an abbreviation).

There are three commands to influence the breaks:

`~` creates a word break that may not be used to split a line.

```
Dr.~Livesey
```

`\␣` creates a word break.

```
\LaTeX\␣is␣a␣good␣thing.
```

`\@` before a punctuation character forces a sentence break.

```
...the␣C.U.S\@.␣But␣some...
```

5.5 Vertical White Space in Output

LaTeX has a very good line-splitting algorithm for plain text. The single line breaks in the input text have no effect on the final output's line breaks.

The `\\` command will force a line break. The `\\*` command will force a line break but not allow it to be used for a page break. (c.f. the use of `~` for a non-splitting horizontal break.)

A blank line (including lines consisting solely of whitespace) in the input will cause a paragraph break and the `\newpage` command will force a page throw.

6 Text

This is where we come to the meat of LaTeX. We will go through the most useful facilities it has to offer but bear in mind that it has many more. See Lamport's book for examples.

6.1 Sectioning

LaTeX provides tools for splitting a document into parts. The most commonly used are the `\section`, `\subsection`, and `\subsubsection` commands. For example:

```
\section{First lump}
\subsection{First bit of first lump}
\subsection{Second bit of first lump}
```

gives

1 First lump

1.1 First bit of first lump

1.2 Second bit of first lump

LaTeX keeps track of the various numbers needed for the sections and so on.

This information can be used to generate a table of contents with the imaginatively named `\tableofcontents` command. However, see section 8 on why this may require running LaTeX *twice* or more.

All the sectioning commands take an optional argument. If present, the optional argument gives the entry for the table of contents which can, therefore, be distinct from the entry in the document itself.

6.2 Titles

There are two components to setting up the title of a document: all the definitions about a document (title, author, date) are defined in the preamble. To actually generate the title the `\maketitle` command is used in the body of the document, usually its first line.

The three basic definition commands for the preamble are:

`\title{`*title*`}`

`\author{`*authors*`}`

`\date{`*date*`}`

If the date is omitted the date on which the `latex` command was run is used. For example

```
\title{\LaTeX}
\author{Bob Dowling}
\begin{document}
\maketitle
\end{document}
```

gives

LaTeX

Bob Dowling

October 21, 1994

Special, titlepage footnotes can be attached to the title or authors with the `\thanks` command. The macro was designed for acknowledgements but is used more widely. For example:

```
\title{\LaTeX}
\author{Bob Dowling\thanks{rjd4@ucs.cam.ac.uk}}
\maketitle
```

gives

LaTeX

Bob Dowling[a]

October 21, 1994

[a]rjd4@ucs.cam.ac.uk

If there are multiple authors they can be listed in the argument of the `\author` command, delimited by `\and`. They can each have their own `\thanks` added. For example:

```
\title{\LaTeX}
\author{Bob Dowling\thanks{rjd4@ucs.cam.ac.uk}
\and S.P.Q. Romanus\thanks{spqr1}}
\maketitle
```

gives

LaTeX

Bob Dowling[a] S.P.Q. Romanus[b]

October 21, 1994

[a]rjd4@ucs.cam.ac.uk
[b]spqr1

6.3 Footnotes

Footnotes are created (and their numbers maintained) by the `\footnote` command as illustrated below:

```
``Five!'' cried the captain.  Come, that's
better.  Five against three leaves us four to
nine.  That's better odds than we had at
starting.  We were seven to nineteen then, or
thought we were, and that's as bad to
bear.''\footnote{The mutineers were soon only
eight in number, for the man shot by
Mr.~Trelawney on board the schooner died that
same evening of his wound.}
```

"Five!" cried the captain. Come, that's better. Five against three leaves us four to nine. That's better odds than we had at starting. We were seven to nineteen then, or thought we were, and that's as bad to bear."[a]

[a]The mutineers were soon only eight in number, for the man shot by Mr. Trelawney on board the schooner died that same evening of his wound.

6.4 Marginal notes

In addition to notes at the foot of the page, LaTeX can put them in the margin. The `\marginpar` command puts a note in the margin with the first line of the note level with the line in the text where the command appeared. For example:

```
``No, not I,'' said Silver.  ``Flint was cap'n; I
was quartermaster, along of my timber leg.  The
same broadside I lost my leg, old Pew lost his
deadlights.  \marginpar{Loss of Silver's leg
described here.} It was a master surgeon, him
that ampytated me---out of college and
all---Latin by the bucket, and what not; but he
was hanged like a dog, and sun-dried like the
rest at Corso Castle.''
```

gives

> "No, not I," said Silver. "Flint was cap'n; I was quartermaster, along of my timber leg. The same broadside I lost my leg, old Pew lost his deadlights. It was a master surgeon, him that ampytated me—out of college and all—Latin by the bucket, and what not; but he was hanged like a dog, and sun-dried like the rest at Corso Castle."

Loss of Silver's leg described here.

6.5 Emphasis

If you want to emphasize a section of text, the command `\emph` changes the font of the text in its argument. If the ambient font is the standard one (roman) then the text in the argument is output in italics. If the ambient font is italic then the emphasized text is set in roman. (See section 10 for details on how to change font.) The following is a simple example:

```
I could not doubt that this was the
\emph{black spot}; and taking it up, I found
written on the other side, in a very good,
clear hand, this short message: ``You have
till ten to-night.''
```

I could not doubt that this was the *black spot*; and taking it up, I found written on the other side, in a very good, clear hand, this short message: "You have till ten to-night."

7 Environments

Most LaTeX commands can only be given between the `\begin{document}` and `\end{document}` commands. This process can be taken further. LaTeX provides functions called "environments" which are a pair of commands `\begin{`*thing*`}` and `\end{`*thing*`}` between which the formatting of the text is subject to different rules and extra commands are available. Many of LaTeX's facilities are provided via environments.

7.1 Quotations

Quotations are typically indented on either side to make them stand out from the main running text. This is achieved by setting up a "quotation environment". The quotation is placed between `\begin{quotation}` and `\end{quotation}`.

```
Over on the back the same hand had written
this further information:---

\begin{quotation}
``Tall tree, Spy-glass shoulder, bearing a
point to the N.  or N.N.E.

``Skeleton Island E.S.E. and by E.

``Ten feet.

``The bar silver is in the north cache; you
can find it by the trend of the east
hummock, ten fathoms south of the black crag
with the face on it.

``The arms are easy found, in the sand hill.
N. point of north inlet cape, bearing E. and
a quarter N.''

\end{quotation}
That was all; but brief as it was, and to
me, incomprehensible, it filled the squire
and Dr.~Livesey with delight.
```

Over on the back the same hand had written this further information:—

> “Tall tree, Spy-glass shoulder, bearing a point to the N. or N.N.E.
> “Skeleton Island E.S.E. and by E.
> “Ten feet.
> “The bar silver is in the north cache; you can find it by the trend of the east hummock, ten fathoms south of the black crag with the face on it.
> “The arms are easy found, in the sand hill. N. point of north inlet cape, bearing E. and a quarter N.”

That was all; but brief as it was, and to me, incomprehensible, it filled the squire and Dr. Livesey with delight.

7.2 Poetry

Poetry provides another example of a simple environment:

```
\begin{verse}
While Dawn, Day's herald straddling
the whole sky,\\
Offers the drowsy world a toast `To Wine',\\
The Sun spills early gold on city roofs---\\
Day's regal Host, replenishing his jug.

Then shouts ring out among us at the tavern:\\
`Rise too, you good-for-nothing tavern lad!\\
Refill our empty bowls with today's measure\\
Before the measure of our lives be filled!'
\end{verse}
```

This generates the output:

> While Dawn, Day's herald straddling the
>   whole sky,
> Offers the drowsy world a toast 'To Wine',
> The Sun spills early gold on city roofs—
> Day's regal Host, replenishing his jug.
>
> Then shouts ring out among us at the tavern:
> 'Rise too, you good-for-nothing tavern lad!
> Refill our empty bowls with today's measure
> Before the measure of our lives be filled!'

7.3 Lists

So far, the environments we have seen simply cause differences in formatting. Environments have the potential to create new commands that exist within them, but not outside. A simple example of this is given by the three environments for generating lists. Consider the following:

```
\begin{itemize}
\item The first topic is dull
\item The second topic is duller
\begin{itemize}
\item The first subtopic is silly
\item The second subtopic is stupid
\end{itemize}
\item The third topic is dullest
\end{itemize}
```

which generates the following output:

- The first topic is dull
- The second topic is duller
  - ∘ The first subtopic is silly
  - ∘ The second subtopic is stupid
- The third topic is dullest

The `\item` command does not exist except within the `itemize` environment. It takes an optional argument which is the "bullet" to use.

If we replace `itemize` with `enumerate` to give

```
\begin{enumerate}
\item The first topic is dull
\item The second topic is duller
\begin{enumerate}
\item The first subtopic is silly
\item The second subtopic is stupid
\end{enumerate}
\item The third topic is dullest
\end{enumerate}
```

we get

1. The first topic is dull
2. The second topic is duller
   (a) The first subtopic is silly
   (b) The second subtopic is stupid
3. The third topic is dullest

The final list environment of the three provided is `description`. In this environment the optional argument of `\item` is always used:

```
\begin{description}
\item[The first topic] is dull
\item[The second topic] is duller
\begin{description}
\item[The first subtopic] is silly
\item[The second subtopic] is stupid
\end{description}
\item[The third topic] is dullest
\end{description}
```

gives the modified output:

The first topic is dull

The second topic is duller

  The first subtopic is silly

  The second subtopic is stupid

The third topic is dullest

7.4 What you see is what you get!

It is sometimes necessary to reproduce exactly what is typed, with no interpretation being done by LaTeX. A good example of this is the writing of LaTeX documentation! LaTeX provides two mechanisms for doing this: a command for exactly reproducing a small amount of data in-line, and an environment for exactly reproducing, and displaying, a larger amount of data.

The commands `\verb` and `\verb*` do not take an argument passed in braces like most other LaTeX commands, but instead look at the next character passed. Everything between that character and the next occurrence of that character is duplicated exactly, without any interpretation by LaTeX. The `\verb*` command renders spaces as "␣" rather than as " ":

- `The \verb!\LaTeX! command gives the ''\LaTeX'' logo.`

  The `\LaTeX` command gives the "LaTeX" logo.

The `verbatim` environment will render whole chunks of text without LaTeX processing:

```
\begin{verbatim}
Line
breaks in the input
are respected and
look at all these weird characters:
!@#$%^&*()_+|~-=\`[]{};':"<>/?
\end{verbatim}
```

```
Line
breaks in the input
are respected and
look at all these weird characters:
!@#$%^&*()_+|~-=\`[]{};':"<>/?
```

Just as the `\verb*` command renders spaces differently from `\verb`, there is a `verbatim*` environment that renders spaces as "␣".

7.5 Tabular data

A common demand is for the display of tabulated data and LaTeX provides the `tabular` environment for this purpose. For example the input

```
\begin{tabular}{|l|c|r|}
\hline
abcdef & abcdef & abcdef \\
abcde & abcde & abcde \\
abcd & abcd & abcd \\
\hline
abc & abc & abc \\
ab & ab & ab \\
a & a & a \\
\hline
\end{tabular}
```

generates the following output:

abcdef	abcdef	abcdef
abcde	abcde	abcde
abcd	abcd	abcd
abc	abc	abc
ab	ab	ab
a	a	a

- The `\begin{tabular}` command takes an argument. It is the first environment we have seen that does.
- The argument `{|l|c|r|}` specifies the alignment of the contents of the various columns:

  l: left aligned,

  c: centre aligned,

  r: right aligned.
- The vertical bars indicate where the vertical lines of the table are to be inserted. They are not compulsory; omitting them simply gives a table without its vertical lines.
- The `\\` command is used to indicate the end of the line.
- (The final line does *not* end with `\\`.)
- The `&` is used to delimit entries within a line.
- The `\hline` command gives a horizontal line across the table.

Now consider the table:

Userid	UID	Name
root	0	Superuser
rjd4	2049	Bob
mug99	2193	Software Testing
spqr1	No such user	
fjc55		

This was created with the input:

```
\begin{tabular}{|l|r|l|}
\hline
Userid & UID & Name \\
\hline
root & 0 & Superuser \\
\hline
rjd4 & 2049 & Bob \\
\hline
mug99 & 2193 & Software Testing \\
\hline
spqr1 & \multicolumn{2}{l|}{No such user}\\
\cline{1-1}
fjc55 & \multicolumn{2}{l|}{}\\
\hline
\end{tabular}
```

There are two new features in this example:

- `\multicolumn` takes an argument and spreads it over several columns. Its three arguments are, in order:
  1. `{2}` The number of columns to be spanned.
  2. `{l|}` The formatting to be used. The *trailing* vertical bar must be provided here, not the leading bar.
  3. `{No such user}` The text to be placed in the wide column.
- `\cline{1-1}` Places a horizontal line over just those columns in the range it takes as its argument. The argument *must* be a range; even if it ranges over only one value, as in this example.

8 Cross-references

8.1 Cross-referencing within text

Consider the following piece of text:

> The section on cross-references is section 8 and appears on page 36 of the notes.

This text was not generated by typing in the numbers 8 and 36. Instead, *references* to the section were used, and LaTeX evaluated them itself. The section was started with the text

```
\section{Cross-references}\label{xref}
```

and the text above that made the references was created by

```
The section on cross-references is
section~\ref{xref} and appears on
page~\pageref{xref} of the notes.
```

The key to this cross-referencing is in the three commands:

- `\label{`*name*`}` records
  - records the number of the section, subsection, theorem etc. of whatever it follows,
  - records the page number where it occurs, and
  - stores the information so gathered in a file called `wombat.aux` (if the LaTeX source is in `wombat.tex`).
- `\ref{`*name*`}` extracts the corresponding section, subsection, theorem etc. number from the `.aux` file.
- `\pageref{`*name*`}` extracts the corresponding page number from the `.aux` file.

There is, however, a problem.

References can precede the labelling.

To see how LaTeX gets round this problem, we need to see just what it does with a reference. We will start with just `wombat.tex`:

```
\documentclass{article}
\begin{document}
On page~\pageref{koala} reference is
made to the koala\label{koala} bear.
\end{document}
```

1. Run `latex wombat`
2. LaTeX looks (unsuccessfully) for `wombat.aux`.

   ```
   No file wombat.aux.
   ```
3. LaTeX gives a warning for the `\pageref` command:

   ```
   Reference ‘koala’ on page 1 undefined.
   ```
4. LaTeX writes “**??**” for `\pageref{koala}` in the DVI output.
5. LaTeX writes the pagenumber to `wombat.aux` for the `\label` command:

   ```
   \newlabel{koala}{{}{1}}
   ```
6. LaTeX ends with a warning that it couldn't do all the cross-referencing:

   ```
   There were undefined references.
   ```
7. LaTeX suggests that it be rerun:

   ```
   Rerun to get cross-references right.
   ```
8. So we rerun `latex wombat`.
9. LaTeX finds `wombat.aux`.
10. LaTeX writes a page number for the `\pageref{koala}` command.
11. LaTeX overwrites `wombat.aux` for the `\label{koala}` command.

12. LaTeX ends without any warning messages.

Note that the file `wombat.aux` always refers to the page and reference numbers generated by the *previous* run of LaTeX. If labels change while running LaTeX then a warning message is printed at the end:

```
Label(s) may have changed. Rerun to get cross-references right.
```

8.2 Simple Bibliographies

Bibliographies are very similar to cross-references, except that the information being referenced is all together in the references list at the back of the document. Two commands and an environment are provided for this purpose: `\cite`, `\bibitem` and the `thebibliography` environment:

- `\bibitem` is a combination of `\item` and `\label`.
- `\cite` is equivalent to `\ref`.
- `thebibliography` is analogous to `enumerate`.

Consider the following example:

```
In ``Treasure Island''~\cite{rls-ti} Stevenson
created the ultimate boy's adventure.
\begin{thebibliography}{MM}
\bibitem{rls-ti} R.L.~Stevenson, ``Treasure Island''.
\bibitem{rls-k} R.L.~Stevenson, ``Kidnapped''.
\end{thebibliography}
```

which gives the output:

> In “Treasure Island” [1] Stevenson created the ultimate boy's adventure.
>
> **References**
>
> [1] R.L. Stevenson, “Treasure Island”.
>
> [2] R.L. Stevenson, “Kidnapped”.

The argument to the `\begin{bibliography}` command is a string that must be at least as wide as any of the labels used in the bibliography.

The key that is generated by `\bibitem` and printed by `\cite` can be set by the user by an optional argument to `\bibitem`: Consider the following example:

```
In ``Treasure Island''~\cite{ti} Stevenson created
the ultimate boy's adventure.
\begin{thebibliography}{MMMM}
\bibitem[TrIs]{ti} R.L.~Stevenson, ``Treasure Island''.
\bibitem[Kidn]{k} R.L.~Stevenson, ``Kidnapped''.
\end{thebibliography}
```

which gives the output:

> In “Treasure Island” [TrIs] Stevenson created the ultimate boy’s adventure.
>
> **References**
>
> [TrIs] R.L. Stevenson, “Treasure Island”.
>
> [Kidn] R.L. Stevenson, “Kidnapped”.

If you wish to pass a note with the citation, a reference to a particular chapter, for example, this can be done by passing an optional argument to the `\cite` command. Consider:

```
In his ``shore adventure''~\cite[Part~3]{rls-ti},
```

which gives:

> In his “shore adventure” [1, Part 3],

Again, these citations are written to the `.aux` file which reflects the *previous* run of LaTeX. LaTeX must be rerun until the labels are consistent.

If you want to refer to a shared bibliography, or make more advanced use of bibliographies, you should consider using the BibTeX command described in appendix B of Lamport's book.

9 Floating blocks

Often with diagrams, figures and tables, it does not matter exactly where they are in the document, so long as they are near to the point where they are first referenced. These are referred to as “floating blocks”. These typically display something horizontally centred with a caption beneath it.

LaTeX provides a couple of almost identical environments for this: `table` and `figure`. We will consider `figure` here first.

Suppose we have something we want to display:

```
\begin{center}
\begin{tabular}{|c|c|}
\hline
A&B\\
\hline
C&D\\
\hline
\end{tabular}
\end{center}
```

A	B
C	D

but which we want to “float” and carry the caption “The ABCD diagram”. We wrap the table’s definition (everything from the `\begin{tabular}` to the `\end{tabular}` inclusive) in a `figure` environment and insert a caption with the `\caption` command:

```
\begin{figure}[hb]
\begin{center}
\begin{tabular}{|c|c|}
\hline
A&B\\
\hline
C&D\\
\hline
\end{tabular}
\end{center}
\caption{The ABCD Diagram}
\end{figure}
```

to give the table that should appear somewhere near here!

A	B
C	D

Figure 1: The ABCD Diagram

The optional argument to the `\begin{figure}` command is a list of acceptable placings in order of preference:

- `h`: *H*ere.
- `t`: At the *t*op of a page.
- `b`: At the *b*ottom of a page.
- `p`: On a *p*age dedicated to floating blocks.

If this locator is missing, a default of `[tbp]` is used.

Note that the caption was automatically numbered. If the `\caption` command is followed by a `\label` command, then the number that a corresponding `\ref` will refer to is the figure number.

A list of figures can be generated with the `\listoffigures` command. The string used to identify the figure will be the caption unless an optional argument is passed to `\caption`. (Compare this with the optional argument passed to `\section` and its effect on `\tableofcontents` described in section 6.1.)

For example:

```
\begin{figure}[hb]
\begin{center}
\begin{tabular}{|c|c|}
\hline
W&X\\
\hline
Y&Z\\
\hline
\end{tabular}
--- This is figure~\ref{wxyz} on
page~\pageref{wxyz}.
\end{center}
\caption[The WXYZ Diagram]{Something similar
to the ABCD diagram but using the letters
W,X,Y,Z}
\label{wxyx}
\end{figure}
```

W	X
Y	Z

— This is figure 2 on page 43.

Figure 2: The ABCD diagram but using the letters W,X,Y,Z

If we now force the list of figures to be printed with the `\listoffigures` command we get:

List of Figures

As with all cross-referencing in LaTeX this requires (at least) two runs of the program. The information is stored in a file called `wombat.lof` which is written by `\end{document}` if a `\listoffigures` command was found during processing.

All the preceding discussion has been for the `figure` environment. There is another environment called `table` (not to be confused with `tabular`) that is identical to `figure` except that it maintains a separate count (for the purpose of numbering figures/tables) and has a few commands with different names:

tables	figures
`\begin{table}`	`\begin{figure}`
`\end{table}`	`\end{figure}`
`\listoftables`	`\listoffigures`
`wombat.lot`	`wombat.lof`

10 Changing things

Do not change things unless you have to!

LaTeX is an *intensional* language so you should give instructions along the lines of "change to the suitable font for emphasis" rather than "change to italics". As we saw in section 6.5, the way to change font for emphasis is with the `\emph` command:

```
But, by thunder! if it don't make me cold
inside to think of Flint.  This is one of
\emph{his} jokes, and no mistake.
```

But, by thunder! if it don't make me cold inside to think of Flint. This is one of *his* jokes, and no mistake.

Now, if the final printer of the document being provided decides that emphasized text should be underlined, rather than italicized, this can be done simply by changing the definition of `\emph`, and leaving the body of the text unchanged:

But, by thunder! if it don't make me cold inside to think of Flint. This is one of <u>his</u> jokes, and no mistake.

10.1 Defining new commands

This is all very well for emphasis, but what if you need some distinguishing feature for something other than emphasis? For example, in the text of "Treasure Island" there is much reference to ships' names. In the copy of the book the author owns, the names are in italics. The names might also be wanted in roman script but within quotation marks. We will start with an example of a `\shipname` command which will simply enclose its arguments in quotation marks. The following definition goes in the *preamble*:

```
\newcommand{\shipname}[1]{``{#1}''}
```

Let us consider that command, piece by piece:

`\newcommand`: This is the command that introduces a new command. It cannot be used to change an existing command.

`\shipname`: `\newcommand`'s first argument is the name of the command being created.

`[1]`: `\newcommand` takes an optional argument which is the number of arguments the new command will take. The default value is zero.

`‘‘{#1}’’`: The final argument is the text that will be substituted for `\shipname{`*ship*`}`. `#1` will be replaced with the first argument passed to `\shipname`.

Braces are placed around `#1` as a precaution. If a command is passed within the argument to `\shipname` that would effect the behaviour of text beyond it, then this causes the change to be limited to within the brackets.

The result of this command is that we can now enter text like the following:

```
The \shipname{Hispaniola} was under her
mainsail and two jibs, and the beautiful
white canvas shone in the sun like snow or
silver.
```

to get output like this:

> The “Hispaniola” was under her mainsail and two jibs, and the beautiful white canvas shone in the sun like snow or silver.

10.2 Changing Fonts

The fonts that LaTeX has at its disposal are parameterized by three values:

The family: By default, you get Computer Modern Roman. Alternatives available usually are Computer Modern Sans (a *sans serif* font) and Computer Modern Typewriter.

The series: This specifies the width and weight (“boldness”). The default is a medium series with boldface also available.

The shape: The default shape is "upright". Alternatives typically available include italics, slanted and "small caps".

These three parameters are meant to be orthogonal; you can change the family, series and shape independently if your system has all the various combinations available to it. But typically it won't. For example, consider the following tables generated on a standard LaTeX installation:

Roman	Medium series	Bold face
Upright	The quick brown fox	**The quick brown fox**
Italic	*The quick brown fox*	***The quick brown fox***
Slanted	*The quick brown fox*	***The quick brown fox***
Small caps	THE QUICK BROWN FOX	**The quick brown fox**
Sans serif		
Upright	The quick brown fox	**The quick brown fox**
Italic	*The quick brown fox*	**The quick brown fox**
Slanted	*The quick brown fox*	**The quick brown fox**
Small caps	THE QUICK BROWN FOX	**The quick brown fox**
Typewriter		
Upright	`The quick brown fox`	`The quick brown fox`
Italic	*`The quick brown fox`*	*`The quick brown fox`*
Slanted	*`The quick brown fox`*	`The quick brown fox`
Small caps	`THE QUICK BROWN FOX`	`The quick brown fox`

As can be seen, LaTeX has failed to provide some of the bold face combinations, and has attempted a "best fit" in these cases.

And now to the meat of the matter: the commands for modifying fonts are illustrated in table 6.

So, if I wanted my ship names to be in italics, I would use the line

```
\newcommand{\shipname}[1]{\textit{#1}}
```

in the preamble and all the shipnames flagged by `\shipname` would be changed to the new style of having italics names, rather than the old style of quoted names.

These commands can be "stacked". If I want bold italics, I can request `\textbf{\textit{TEXT}}` or `\textit{\textbf{TEXT}}`.

`\textrm{Roman family}`	Roman family
`\textsf{Sans serif family}`	Sans serif family
`\texttt{Typewriter family}`	`Typewriter family`
`\textup{Upright shape}`	Upright shape
`\textit{Italic shape}`	*Italic shape*
`\textsl{Slanted shape}`	*Slanted shape*
`\textsc{Small caps shape}`	SMALL CAPS SHAPE
`\textmd{Medium series}`	Medium series
`\textbf{Boldface series}`	**Boldface series**

Table 6: Font modifying commands

Explicit font changing has no place in the body of the document. You should decide *why* you want to change font, and then create a new command that deals with that specific reason. If you want to italicize ship names, then you should write a `\shipname` command, define it to do italicizing in the preamble, and use `\shipname` throughout the body of the document.

11 Simple Diagrams in LaTeX

LaTeX provides support for simple diagrams in its `picture` environment. Consider diagram 7. This is generated by the source in figure 8.

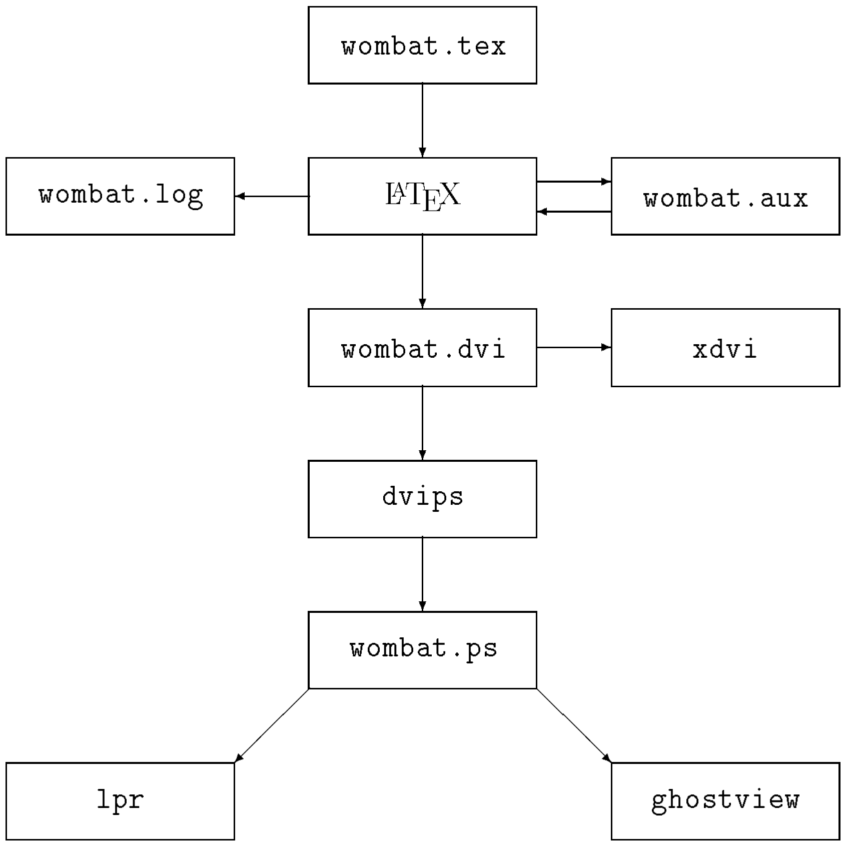


Table 7: An example diagram in LaTeX

```
%\unitlength is set in the preamble (to 1mm)
\begin{picture}(110,110)
% First row
\put(40,100){\framebox(30,10){\file{wombat.tex}}}
\put(55,100){\vector(0,-1){10}}
% Second row
\put(0,80){\framebox(30,10){\file{wombat.log}}}
\put(40,85){\vector(-1,0){10}}
\put(40,80){\framebox(30,10){\LaTeX}}
\put(70,87){\vector(1,0){10}}
\put(80,83){\vector(-1,0){10}}
\put(80,80){\framebox(30,10){\file{wombat.aux}}}
\put(55,80){\vector(0,-1){10}}
% Third row
\put(40,60){\framebox(30,10){\file{wombat.dvi}}}
\put(70,65){\vector(1,0){10}}
\put(80,60){\framebox(30,10){\cmd{xdvi}}}
\put(55,60){\vector(0,-1){10}}
% Fourth row
\put(40,40){\framebox(30,10){\cmd{dvips}}}
\put(55,40){\vector(0,-1){10}}
% Fifth row
\put(40,20){\framebox(30,10){\file{wombat.ps}}}
\put(40,20){\vector(-1,-1){10}}
\put(70,20){\vector(1,-1){10}}
% Sixth row
\put(0,0){\framebox(30,10){\cmd{lpr}}}
\put(80,0){\framebox(30,10){\cmd{ghostview}}}
\end{picture}
```

Table 8: The source for diagram 7

We will now consider this in detail as an illustration of how to create diagrams in LaTeX.

11.1 Scaling

All sizes and lengths in `picture` mode are given in terms of `\unitlength`. This is typically set in the preamble with a command like

```
\setlength{\unitlength}{10mm}
```

11.2 Size of picture

The `picture` environment is started with the command

`\begin{picture}`(X,Y)

The *(X,Y)* coordinate pair defines the size of the picture (in terms of the `\unitlength`). In all the subsequent examples, a box will be drawn to indicate the perimeter of the picture and to identify the origin (bottom left).

11.3 Placing objects in a diagram

The fundamental command of the `picture` environment is the `\put` command:

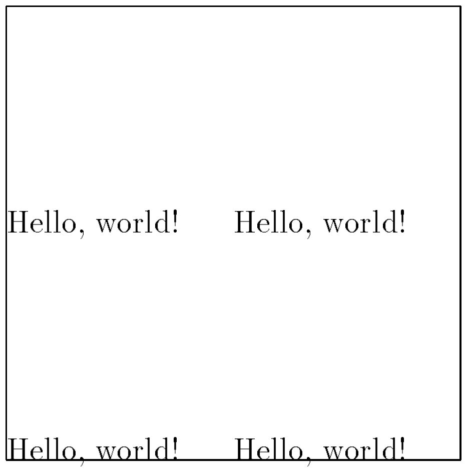
`\put(`*x,y*`){`*object*`}`

- `\put` is the fundamental command of the `picture` environment.
- The *(x,y)* defines the location of the object being placed, with respect to the coordinate origin of the `picture` environment.
- The *object* is a "sub-picture" made from characters of a font designed for this environment.
- Because of this, there are a limited number of shapes and slopes available.

To place text, simply pass it as the graphical argument to `\put`. The coordinates in `\put` are the lower left corner of the text. For example:

```
\begin{picture}(60,60)
\put(0,0){Hello, world!}
\put(30,0){Hello, world!}
\put(0,30){Hello, world!}
\put(30,30){Hello, world!}
\end{picture}
```

gives[2]

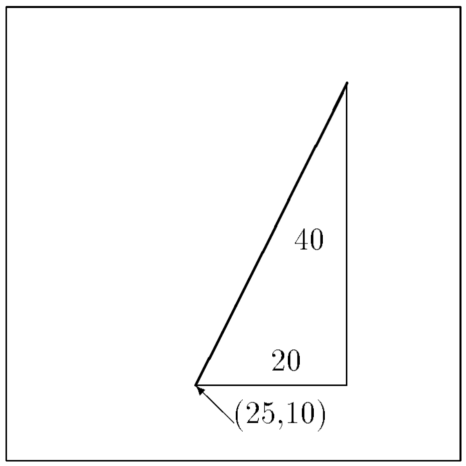


11.4 Lines and Arrows

Consider the command

```
\put(25,10){\line(1,2){20}}
```

This generates the line shown below in bold:



The things to note about this are:

[2]Remember that the frame is being added artificially.

- `\put(25,10)` determines where the line should start.
- `(1,2)` determines the *direction* of the line but *not* its length.
- `{20}` determines the extent of the line in the x-direction.

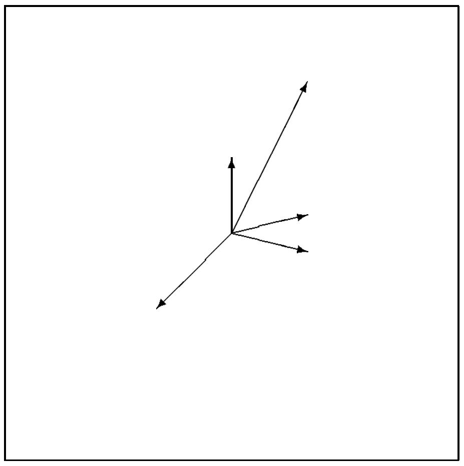
The line is constructed from a font of slopes, so only a discrete set of gradients is available. The rules for determining what slopes are available is as follows. Consider the *(x,y)* pair:

- x and y must lie in the range -6 to $+6$ inclusive.
- If either is zero, the other must be ± 1.
- Otherwise, x and y must be coprime.
- If x is 0 then the length parameter measures the y-direction.

There are 25 slopes available.

To get arrows, simply use the `\vector` command rather than the `\line` command. The slope numbers can only range from -4 to $+4$:

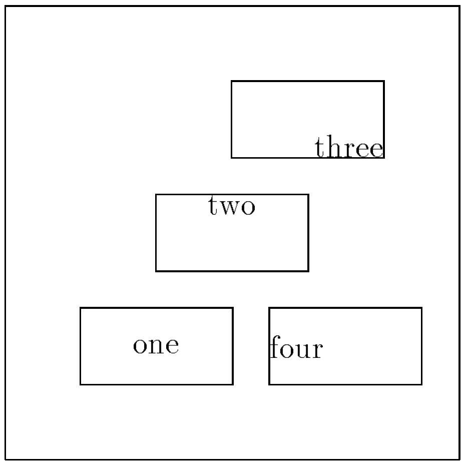
```
\begin{picture}(60,60)
\put(30,30){\vector(0,1){10}}
\put(30,30){\vector(1,2){10}}
\put(30,30){\vector(-1,-1){10}}
\put(30,30){\vector(4,-1){10}}
\end{picture}
```



11.5 Boxed Text

Putting a box around text is done by the `\framebox` command. `\framebox` takes a coordinate argument to determine the dimensions of the box, and an optional argument to control the justification of the text within the box:

```
\put(10,10){\framebox(20,10){one}}
\put(20,25){\framebox(20,10)[t]{two}}
\put(30,40){\framebox(20,10)[br]{three}}
\put(35,10){\framebox(20,10)[l]{four}}
```



11.6 Conclusion

The `picture` environment is vastly more powerful than has been described here. Refer to chapter 7 of Lamport's book for more details.

12 Mathematics

LaTeX is based on TeX, a typesetting package that was originally designed for use with mathematics. LaTeX inherits from TeX some very powerful mathematical typesetting facilities. The spacing rules in mathematics are entirely different from those in the text we have seen so far.

Displayed maths comes in two principal varieties, "in-line" and "displayed". In-line maths leads to expressions of the form:

> Einstein derived $\Delta E = \Delta m \, c^2$ from the more important expression $|p|^2 = m_0^2 \, c^2$.

whereas displayed maths appears as:

> Einstein derived
> $$\Delta E = \Delta m \, c^2$$
> from the more important expression
> $$|p|^2 = m_0^2 \, c^2.$$

There are three distinct ways to generate in-line maths:

- `\(\Delta E = \Delta m \, c^2\)`
- `$\Delta E = \Delta m \, c^2$`
- `\begin{math}`

  `\Delta E = \Delta m \, c^2`

  `\end{math}`

There are two ways to generate displayed maths:

- `\[\Delta E = \Delta m \, c^2\]`
- `\begin{displaymath}`

  `\Delta E = \Delta m \, c^2`

```
    \end{displaymath}
```

The author's preferences are for the bracket forms, `\(...\)` and `\[...\]`.

If the environment `equation` is used in place of `displaymath` then, in addition to being displayed, the equation is numbered. Use of `\label` in this environment will set up references to the equation number. The evaluation of the equation numbers is done automatically, and the nature of the numbers assigned is determined by the style.

By default, displayed mathematical expressions are centred horizontally and equation numbers are placed at the right hand side of the page. There are two options that can be passed to the standard documentclasses, `leqno` and `fleqn`. `leqno` causes equation numbers to be on the left hand side of the page and `fleqn` causes displayed maths to be aligned so that it starts a fixed distance from the left margin.

12.1 Mathematical symbols

First we must note that the spacing around a symbol in maths mode depends on the character's mathematical status.

Simple variables (such as x) get minimal space around them,

Binary operations (such as the addition operator, $+$) get slightly more,

Relations (such as equality, $=$) get most.

It is sometimes necessary to help out LaTeX with the spacing and this will be covered later in section 12.5.

Table 9 gives the commands for generating the Greek alphabet. These are all simple variables. Note that some characters have multiple versions. For example `\epsilon` generates ϵ and `\varepsilon` generates ε. Also note that where a letter of the Greek alphabet is identical to a Latin letter, the Latin letter should be used. So lowercase omicron is given by `o` and uppercase beta by `B`.

Some characters are not available by default and require the `latexsym` package, loaded in by the `\usepackage{latexsym}` command in the preamble.

It isn't worth remembering which characters are in the default LaTeX and which are in the extra package, so the author recommends *always* loading the `latexsym` package in a mathematical document, even if it doesn't end up being used. For even more bizarre characters, the `amssymb` package may be available. The "LaTeX Companion" has details in chapter 8.

A limited set of accents are available in maths mode. These are given in table 10. Unlike text accents, these can be stacked fairly freely.

The binary operations are listed in table 11. It is impossible to create a complete set of mathematical binary operations; mathematicians take a perverse delight in coming up with new notations to annoy the most conscienscious of typesetters.

Binary relations are given in table 12. Note that it is possible to negate any binary relation by preceding it with a `\not`. So `\equiv` gives $\equiv$ and `\not\equiv` gives $\not\equiv$.

LaTeX's set of arrows is given in table 13. These have similar spacing to relations. Finally, the miscellanea are given in table 14.

12.2 Super- and subscripts

In maths mode the circumflex, "`^`", superscripts the next character or group of characters, and the underscore, "`_`", subscripts it. A fairly exhaustive set of examples is given in table 15.

It is important to remember that if more that one character is to be scripted then they must be grouped in braces.

LaTeX has two levels of scripting, "script" and "script script". In an expression like A^{BC} (`\(A^{BC}\)`), the B and C are both superscripts of A. However, in an expression like A^{B^C} (`\(A^{B^C}\)`) the C is a superscript of B which is, in turn, a superscript of A; C is a supersuperscript. The expressions "`A^B^C`" and "`A_B_C`" are ambiguous and will cause an error. Also note the difference between A^B_C (`\(A^B_C\)` or `\(A_C^B\)`) and ${A^B}_C$ (`\({A^B}_C\)`).

`\alpha`	α	`\beta`	β	`\gamma`	γ	`\delta`	δ
`\epsilon`	ϵ	`\varepsilon`	ε	`\zeta`	ζ	`\eta`	η
`\theta`	θ	`\iota`	ι	`\kappa`	κ	`\lambda`	λ
`\mu`	μ	`\nu`	ν	`\xi`	ξ	`o`	o
`\pi`	π	`\varpi`	ϖ	`\rho`	ρ	`\varrho`	ϱ
`\sigma`	σ	`\varsigma`	ς	`\tau`	τ	`\upsilon`	υ
`\phi`	ϕ	`\varphi`	φ	`\chi`	χ	`\psi`	ψ
`\omega`	ω						
`A`	A	`B`	B	`\Gamma`	Γ	`\Delta`	Δ
`E`	E	`Z`	Z	`\Theta`	Θ	`H`	H
`I`	I	`K`	K	`\Lambda`	Λ	`M`	M
`N`	N	`\Xi`	Ξ	`O`	O	`\Pi`	Π
`P`	P	`\Sigma`	Σ	`\Upsilon`	Υ	`\Phi`	Φ
`X`	X	`\Psi`	Ψ	`\Omega`	Ω		

Table 9: Letters of the Greek alphabet

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

Table 10: The set of maths accents

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

Table 11: Binary operations

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

Table 12: Binary relations

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

Table 13: Arrows

`\aleph`	$\aleph$	`\forall`	$\forall$	`\exists`	$\exists$
`\ell`	ℓ	`\imath`	$\imath$	`\jmath`	$\jmath$
`\prime`	$\prime$	`\emptyset`	$\emptyset$	`\hbar`	$\hbar$
`\Box`	$\Box$	`\triangle`	$\triangle$	`\Diamond`	$\Diamond$
`\clubsuit`	$\clubsuit$	`\spadesuit`	$\spadesuit$	`\infty`	∞
`\heartsuit`	$\heartsuit$	`\diamondsuit`	$\diamondsuit$	`\partial`	∂
`\nabla`	∇	`\neg`	$\neg$	`\surd`	$\surd$
`\flat`	$\flat$	`\natural`	$\natural$	`\sharp`	$\sharp$
`|`	$\Vert$	`\top`	$\top$	`\bot`	$\bot$
`\wp`	$\wp$	`\Re`	$\Re$	`\Im`	$\Im$
`\mho`	$\mho$	`\angle`	$\angle$	`\backslash`	$\backslash$

Table 14: Mathematical miscellany

`A^B`	A^B	`A_B`	A_B
`A^BC`	A^BC	`A_BC`	A_BC
`A^{BC}`	A^{BC}	`A_{BC}`	A_{BC}
`A^{B^C}`	A^{B^C}	`A_{B_C}`	A_{B_C}
`{A^B}^C`	${A^B}^C$	`{A_B}_C`	${A_B}_C$
`A^B^C`	*error*	`A_B_C`	*error*
`A^B_C`	A^B_C	`A_B^C`	A_B^C
`A^{B_C}`	A^{B_C}	`A_{B^C}`	A_{B^C}
`{A^B}_C`	${A^B}_C$	`{A_B}^C`	${A_B}^C$

Table 15: Super- and subscripts

12.3 Other common constructions

Some other commonly used mathematical constructions are illustrated here by example:

12.3.1 Surds

```
\( 2 = \sqrt{4} = \sqrt[4]{16} \)
```

$2 = \sqrt{4} = \sqrt[4]{16}$

12.3.2 Fractions

```
\( \frac{a+bi}{c+di} \)
```

$\frac{a+bi}{c+di}$

NB: The numerator and denominator are cast into script size.

12.3.3 Ellipsis

```
\( a,b,\ldots,y,z,
   a+b+\cdots+y+z \)
```

$a, b, \ldots, y, z, a + b + \cdots + y + z$

Vertical and diagonal dots are available in addition to low and centred horizontal dots. `\ldots` gives low dots, `\cdots` gives centred dots, `\vdots` gives vertical dots and `\ddots` gives diagonal dots, with the diagonal running from top left to bottom right.

12.3.4 Arrays

```
\(
\begin{array}{cccc}
A_{11} & A_{12} & \cdots & A_{1n} \\
A_{21} & A_{22} & \cdots & A_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
A_{m1} & A_{m2} & \cdots & A_{mn}
\end{array}
\)
```

$\begin{array}{cccc} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{array}$

Note that, as in `tabular` mode, the final line does not have a trailing `\\`.

12.4 Symbols that vary according to context

LaTeX maths mode has some facilities for varying the display of characters based on context.

Some symbols change their size *and behaviour* according to whether they are in displayed maths or in-line maths. Consider `\sum_{i=0}^\infty x_i^2` in these two contexts:

- In-line:
  - `\(\sum_{i=0}^\infty x_i^2\)`
  - $\sum_{i=0}^\infty x_i^2$
- Displayed:
  - `\[\sum_{i=0}^\infty x_i^2\]`
  - $$\sum_{i=0}^\infty x_i^2$$

- Different size of sigma
- Different location of limits
- Note that `\sum` is *not* the same as `\Sigma`!

A set of these resizing symbols is given in table 16.

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

Table 16: Symbols that resize according to in-line or display mode

The other set of symbols that need to change their sizes according to context are the delimiters. The most common of these symbols are the parenteses (round brackets). These need to be of a size suitable for whatever it is that they delimit. For example, simple parentheses around an array (perhaps in an attempt to create a matrix) give daft results:

- Source:

```
\[ (
\begin{array}{cc}
A & B \\
C & D
\end{array}
) \]
```

- Output:

$$(\begin{array}{cc} A & B \\ C & D \end{array})$$

- *Wrong!*

Clearly we need a mechanism for causing the delimiters to resize. This is done with the `\left` and `\right` commands:

- Source:

```
\[ \left(
\begin{array}{cc}
A & B \\
C & D
\end{array}
\right) \]
```

- Output:

$$\left(\begin{array}{cc} A & B \\ C & D \end{array}\right)$$

- *Correct!*

The set of delimiters is given in table 17. The delimiters are resized to fit around the maths between the `\left` and the `\right`. The two delimiters used need not match. The uses of `\left` and `\right` must balance. There is an extra "invisible delimiter" generated by "`\left.`" and "`\right.`". This generates no output but can be used to match a delimiter being resized with the other of the `\left`, `\right` pair. Note that it is perfectly acceptable to use `\left)` or `\right(`.

In the author's opinion, the `\left` and `\right` commands should be used with *every* pair of delimiters in a maths text.

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

Table 17: The set of resizable delimiters in LaTeX

12.5 Spacing in mathematical formulas

While text has relatively straightforward spacing rules, mathematical expressions are vastly more complicated to get right automatically. So LaTeX doesn't try too hard; it has some simple rules and allows the user to make modifications to taste. To this end, the spacing commands in table 18 are provided.

thin *negative* space	`\!`
thin space	`\,`
medium space	`\:`
thick space	`\;`
interword space	`\␣`

Table 18: Spacing commands for maths mode

An example of where thin space would be used is in the expression $\Delta E = \Delta m\, c^2$. LaTeX does not know that `\Delta` is supposed to bind closely to `E` and `m`. It cannot distinguish it from an equation involving the product of a variable called Δ and a variable called E (or m). So there are three principal ways to resolve this:

1. Ignore the problem:
   - `\(\Delta E = \Delta m c^2\)`
   - $\Delta E = \Delta m c^2$
   - This relies on context; a mathematician reading this will not make the mistake of treating Δ as a variable.
2. Pull in the Δ to its arguments:
   - `\(\Delta\! E = \Delta\! m c^2\)`
   - $\Delta\! E = \Delta\! m c^2$
   - In the author's *subjective* opinion this looks a bit cramped.
3. Push the Δm construct away from the c^2 construct:
   - `\(\Delta E = \Delta m\, c^2\)`
   - $\Delta E = \Delta m\, c^2$
   - In the author's *subjective* opinion this is the best of the three.

Note that we have been forced to make typesetting decisions. We are now pushing LaTeX hard enough that it is starting to fail in its design aim of removing the need for typesetting skills from the user.

Another example of the need to use explicit spacing in LaTeX is in expressions involving double integrals. Compare the following:

- `\int \int z dx dy`
- $\int \int z dx dy$
- `\int \!\! \int z \, dx \, dy`
- $\int \!\! \int z \, dx \, dy$

12.6 Fonts

Just as text mode has commands for changing font (section 10.2), maths mode has something similar. Table 19 illustrates the commands available. Note that only the Latin characters and the upper case Greek characters have been altered.

In addition to the standard fonts there is also a "caligraphic" font available for capital (Latin alphabet) letters only. The caligraphic alphabet looks like this: $\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$. Note that applying the corresponding font command, `\mathcal`, to anything other than these 26 characters will produce unpredictable results.

`\mathrm{\Psi=\psi+X^2+y^2}`	$\mathrm{\Psi=\psi+X^2+y^2}$
`\mathsf{\Psi=\psi+X^2+y^2}`	$\mathsf{\Psi=\psi+X^2+y^2}$
`\mathtt{\Psi=\psi+X^2+y^2}`	$\mathtt{\Psi=\psi+X^2+y^2}$
`\mathit{\Psi=\psi+X^2+y^2}`	$\mathit{\Psi=\psi+X^2+y^2}$
`\mathbf{\Psi=\psi+X^2+y^2}`	$\mathbf{\Psi=\psi+X^2+y^2}$
`\mathcal{\Psi=\psi+X^2+y^2}`	$\ominus=\psi+\mathcal{X}^{\in}+\dagger^{\in}$

Table 19: Maths mode font modifying commands

Once again, you should not use explicit font changes in the body of your text but should create new commands based on the reason for the change of font. If you want to write all your vectors in a bold face then you should create a new command:

```
\newcommand{\vect}[1]{\mathbf{#1}}
```

Then you can write

```
\( \vect{a} = \vect{b} \times \vect{c} \)
```

and not worry about how the final style is planning to typeset vectors. If the final editor wants to use the "$\vec{a}$" notation then simply changing the line in the preamble to "`\newcommand{\vect}[1]{\vec{#1}}`" will do.

12.7 Aligning equations

So far we have only considered the display of single equations. A common requirement is for multiple equations to be displayed, normally with their equals signs (or other relations) vertically aligned. LaTeX provides a pair of environments called `eqnarray` and `eqnarray*` for this. `eqnarray` will number each equation unless instructed not to for a specific equation. `eqnarray*` will not number its equations. Consider this example of the use of `eqnarray`:

```
\begin{eqnarray}
x & = & y+5-4 \\ \label{x-y}
  & = & y+1 \nonumber \\
  & > & y \label{x-y-inequality}
\end{eqnarray}
```

$$\begin{array}{rcl} x & = & y+5-4 \qquad (1) \\ & = & y+1 \\ & > & y \qquad (2) \end{array}$$

Note that the equations are treated as a three column array with `[rcl]` alignment and that, like other arrays the final line does not end with a `\\`.

12.8 Function names

When a function, such as "sin", is being processed, LaTeX cannot distinguish the sine function from the product of the variables s, i and n. To resolve this, special LaTeX commands are provided for representing the common function names in roman style, rather than the ambient maths italic. For example, the sine function has the LaTeX representation `\sin`.

- Wrong
  - `\( sin^2(\theta) + cos^2(\theta) \equiv 1 \)`
  - $sin^2(\theta) + cos^2(\theta) \equiv 1$
- Right
  - `\( \sin^2(\theta) + \cos^2(\theta) \equiv 1 \)`
  - $\sin^2(\theta) + \cos^2(\theta) \equiv 1$
  - (Actually, we should use `\left(` and `\right)`.)

The complete set of function names defined in LaTeX is given in table 20.

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

Table 20: The set of maths functions names in LaTeX

12.9 Theorems etc.

Mathematics has a huge number of names for statements which are presented in a fairly common manner: theorems, conjectures, postulates, definitions, hypotheses, propositions, lemmas, axioms and many, many more. Furthermore, people's numbering schemes vary widely. Some people have a single numbering scheme so that theorem 1 is followed by definition 2 which is followed by lemma 3 and so on. Other people might group all the statements that need proofs but have separate number tracks for axioms, definitions and hypotheses, so definition 1 would be followed by theorem 1, lemma 2, lemma 3, definition 2, axiom 1 and so on.

Because of this wide variety, the default LaTeX styles define *no* "theorem-style" environments at all, but instead provide a command, `\newtheorem` to create them at the user's behest.

Suppose I want to create an environment for laws (Newton's, Thermodynamics', Murphy's,...) with each law started with the label "Law". I run the command

```
\newtheorem{law}{Law}
```

The first argument to `\newtheorem` is the name of the environment to be created, and the second argument is the label to put at the start of the generated output. If I use the command

```
\begin{law}
The harder you kick it, the faster it accelerates.
\end{law}
```

I get the output:

Law 1 *The harder you kick it, the faster it accelerates.*

and if I follow it with another "law",

```
\begin{law}
You can't push a rope.
\end{law}
```

I get the output

Law 2 *You can't push a rope.*

so a counter is being incremented automatically. If I want to name my laws then I can do so with an optional argument to the `\begin{law}` command:

```
\begin{law}[Murphy]
If it can go wrong, it will go wrong.
\end{law}
```

Law 3 (Murphy) *If it can go wrong, it will go wrong.*

Now suppose I want to define another environment of this style, sharing the same numbering track as the laws are using. It is to be called "`para`" and label its text as "Paradox". To do this, I insert an optional argument in the `\newtheorem` command in the preamble:

```
\newtheorem{para}[law]{Paradox}
```

and then the command

```
\begin{para}
If Murphy's law can go wrong, it will go wrong.
\end{para}
```

Paradox 4 *If Murphy's law can go wrong, it will go wrong.*