Notes of Guide to LaTeX (4 edition)

siecho

yry0204@mail.ustc.edu.cn

2025.10.16 - 2026.02.22

Contents

1 Introduction

This document is a collection of notes taken while studying the book *Guide to LaTeX, 4 edition* by Helmut Kopka and Patrick W. Daly. The notes are organized according to the chapters of the book.

2 Text, Symbols, and Commands

2.1 Command names and arguments

the general syntax of commands:

```
\name[optional]{mandatory}
```

Some commands may have several mandaotry, like:

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

That is ▬

*-form : A * is added to some commands' name to modify their functionality somehow. ex. \section* does not print an automatic section number. like:

Unnumbered Section

However, some commands possess no arguments and are composed of only a name, like \LaTeX produces LaTeX logo, and \TeX → TeX, \LaTeXe → LaTeX 2$_\varepsilon$ as well.

2.2 Environments

An environments affects the text within it by treating it differently according to the env. parameters.

```
\begin{env. name}
...
\end{env. name}
```

For ex., with the quote env.:

```
\begin{quote}
text1 \small text2\ \bfseries text3
\end{quote}
```

text1 text2 **text3**

Most declaration command names may also be used as env. names. For ex., the \em command may be used as the emph env.:

```
\begin{em}
This is an emphasized paragraph.
\end{em}
```

This is an emphasized paragraph.

and the command `\em` *switches to emphasized text until the end of the current group or env. one can recover the normal text by using* `\normalfont` command.

A nameless env. can be simulated by grouping text within curly braces { ... }. The effect of any command within it is limited to the group. One can create own env. as well(section 10.4).

2.3 Declarations

A declaration is a command that changes the formatting of the text that follows it. The preceding **\bfseries** and **\small** are examples of such declarations that alter the current typeface. Some declarations take arguments, like `\fontsize{size}{skip}` as in:

```
\fontsize{14pt}{1pt}\selectfont This is 14pt text with 1pt leading.
```

This is 14pt text with 1pt leading.

and \setlength with two arguments, like:

By using `\setlength{\parindent}{0.5cm}`, the first line of this paragraph is indented by 0.5cm.

Others like:

```
{\centering This text is centered.\\
\newcounter{mycounter} \setcounter{mycounter}{5} My counter value is \themycounter.\\
\setcounter{mycounter}{10} Now my counter value is \themycounter.\\
\addtocounter{mycounter}{3} After adding 3, my counter value is \themycounter.\\
\pagenumbering{roman} Now the page number is in roman numerals: \thepage.\\
\thispagestyle{empty} This page has no page number, \\
and it is not included in the page number counting.\\
\newlength{\mylen} \setlength{\mylen}{2cm} The length is set to \the\mylen.\\
\newsavebox{\mybox} \sbox{\mybox}{This is a saved box.}
The content of the box is: \usebox{\mybox}

}
```

generates:

This text is centered.

My counter value is 5.

Now my counter value is 10.

After adding 3, my counter value is 13.

Now the page number is in roman numerals: i.

This page has no page number,

and it is not included in the page number counting.

The length is set to 56.9055pt.

The content of the box is: This is a saved box.

2.4 Lengths

2.4.1 Fixed lengths

Lengths consist of a number and a unit. The units can be:

- in (inch): 1 in = 2.54 cm
- mm (millimeter)
- cm (centimeter)
- pt (point): 1 in = 72.27 pt
- bp (big point): 1 in = 72 bp
- dd (didot point): 1157 dd = 1238 pt
- cc (cicero): 1 cc = 12 dd
- ex (height of 'x' in the current font)
- em (width of 'M' in the current font)

Both 12,5 cm and 12.5 cm are acceptable. One can also use negative lengths, like -3pt.

2.4.2 Rubber lengths

Rubber lengths consist of a fixed part plus an optional stretch and shrink part. The syntax is:

```
<fixed part> plus <stretch part> minus <shrink part>
```

For ex., the length:

```
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

means that the normal parskip is 1ex, but it can stretch by up to 0.5ex or shrink by up to 0.2ex if necessary.

One special rubber length is \fill, which has the natural length of zero and can stretch infinitely. It is used in commands like \hfill and \vfill to create flexible spaces.

2.5 Special characters

2.5.1 Spaces

The space/blank character is treated specially in LaTeX. During the compilation, blanks are replaced by rubber lengths to allow the line to fill up to the right margin. As a result, some "peculiar" behaviors may occur:

- Multiple consecutive blanks are treated as a single blank.
- A blank at the beginning of a line is ignored.
- A blank after a command name is ignored.
- The end of a line is treated as a blank.

To insert an explicit space, one can use the command \ or \␣.

A protected space, that is, character˜, is used to prevent a line break at that point. For ex., is equivalent to .

2.5.2 Quotation marks

The quotation marks found on the typewriter keyboard are ” or ’ , are not used in LaTeX. Instead, the following commands are used:

- “ (double left backquote)
- ” or ” (double right quote)
- ‘or ‘ (left quote)
- ’or ’ (right quote)

2.5.3 Hyphens and dashes

The character comes in various lengths like:

- — (dash),
- – (en dash), is used in ranges of numbers, like 2023–2024.
- - (hyphen), is used for compound words as father-in-law and for word division at the end of a line.

2.5.4 Printing command characters

The characters # $ & _ % { } are interpreted as commands, so to print them, one must use the \ character before them.

2.5.5 Some special characters

In one’s paper or article, one may used some special characters which do not exist in the typewriter keyboard.

For ex.,

§, †, ‡, ¶, ©, and £

Others like:

œ, Œ, æ, Æ, å, Å, ¡, ¿, ø, Ø, ł, Ł, ß,

2.5.6 Accents

In non-English languages, some special characters are used to represent accents. For ex.,

ò, ó, ô, õ, ō, ȯ, ö, ŏ, ǒ, ő, o͡, o͡o, o, ọ, o̲, o̊, ǫ

2.5.7 The euro symbol

The euro symbol is €. More about it, see section 2.5.8 in textbook.

2.5.8 Typing special symbols directly

The inputenc package takes an encoding parameter (e.g., utf8, latin1) to inform LaTeX: ”The current source file uses this encoding,” allowing the engine to parse characters according to the corresponding rules. If the source file is saved in UTF-8 encoding, one can include the package as follows:

```
\usepackage[utf8]{inputenc}
```

cp437, cp850, applemac, and ansinew are largely obsolete in modern contexts, though they may still appear in legacy systems or very old documents.

2.5.9 Ligatures

Ligatures are special characters that are combined from two or more other characters. For ex., the ligature fi is often used in English to replace the two characters f and i. Others include fl, ff, ffi, and ffl.

Ligatures can be broken by inserting a \/ between the characters, which lead ‘shelfful’ to ‘shelfful’.

2.5.10 The date

In general, The command \today produces the current date in the format YYYY.MM.DD, but in some document classes, it may produce the date in a different format. Such as cteart, which produces the date in the format YYYY 年 MM 月 DD 日. In the head of this file, we redefine the \today command to always produce the date in the format YYYY.MM.DD., by adding the following line to the preamble:

```
\renewcommand{\today}{
\number\year.
\ifnum\month<10 0\fi
\number\month.
\ifnum\day<10 0\fi
\number\day
}
```

2.6 Fine-tuning text

Some unnecessary fine-turning text isn't covered in this chapter, so please refer to section 2.6 in textbook.

2.6.1 Word and character spacing

Sentense termination and periods TeX interprets a period following a lowercase letter as the end of a sentence, but not following an uppercase letter, result in 'Prof. Smith' looking odd. To fix this, one can use \␣ or ~ instead of the normal blank: 'Prof. Smith'.

In addition, ~ is a *protected space* that prevents a line from being broken at that point.

If following an uppercase letter, but one wants to indicate the end of a sentence, one can use `\@` followed by a normal blank.

Inserting arbitrary spaces Spacing with any desired length can be inserted using the commands:

- \hspace{length} : inserts a horizontal space of the specified length.
- \hspace*{length} : inserts a horizontal space of the specified length with *-form, which means it will not be ignored at the beginning or end of a line.

A blank before or after these commands will also be included:

This is   an example of horizontal space.
This is   another example of horizontal space.

The length may be negative, in which case the text following it will move left or up.

The command `\hfill` is an abbreviation of \hspace{\fill}. It inserts a horizontal space that fills the remaining space on the line. Such as:

Left   Right
or
Left   Center   Right

Also, `\quad` and `\qquad` are abbreviations of \hspace{1em} and \hspace{2em}, respectively:

a b c
a b c

Inserting variable ...... and ______ sequence Two commands that work exactly the same as `\hfill` are:

`\dotfill` and `\hrulefill`

Like:

Left .... Right
Left ______________ Center left   Center right .... Right

2.6.2 Line spacing

Using `\\[space]` to insert a vertical space of the specified length between two lines, or `\\*[space]` to do so with *-form to prevent a new page from occurring between the two lines. Howerver, the next line isn't indented.

The command `\newline` works the same way as `\\`, while `\par` inserts a line break with indentation.

`\linebreak[number]` encourages or forces a line break at the point where it is inserted. The optional argument [number] is an integer between 0 and 4 that indicates the desired penalty for the line break. The default value is 2. Its opposite command is `\nolinebreak[number]`.

2.6.3 Vertical spacing

The command \vspace and \vspace* insert vertical spaces, like:

- \vspace{length} : inserts a vertical space of the specified length.
- \vspace*{length} : inserts a vertical space of the specified length with *-form, which means it will not be ignored at the top or bottom of a page.

The command `\vfill` is an abbreviation of \vspace{\fill}. It inserts a vertical space that fills the remaining space on the page. Such as:

Top

Center

Bottom

2.6.4 Page breaks

Normal pages The command `\pagebreak[number]` and `\nopagebreak[number]` are similiar to `\linebreak[number]` and `\nolinebreak[number]`.

The analogy with the command `\linebreak` goes further: Just as the line before the break is left and right justified with extra interword spacing, the page before the break is expanded with interline spacing to make it top and bottom justified.

The proper command to end a page in the middle, fill the rest of the page with whitespace, and start a new page is `\newpage`, which is equivalent to `\newline` with regard to page breaking.

Pages with figures and tables The command `\clearpage` ends the current page and causes all pending floating figures and tables to be printed on one or more extra pages of their own before starting a new page.

Two-column pages If the document class option **twocolumn** has been selected, or the command `\twocolumn` has been issued, the the two commands `\pagebreak` and `\newpage` end the current column and go on a new one. But if one uses package **multicol**, the two commands end the current column, treating the columns as pages. While the command `\clearpage` and `\cleardoublepage` always ends the current page and all pending floating figures and tables will be printed.

Controlling page breaks The command `\enlargethispage{length}` increases the length of the current page slightly.

2.6.5 Word Division

Usually, it often happens that the break cannot be made between words without either shoving the text too close together or inserting huge gaps between the words as a line is to be left and right justified. It is in such cases that LaTeX hyphenates words automatically. Nevertheless, sometimes LaTeX cannot find a suitable place to hyphenate a word, which need to be specified by

1. manually using the command `-\linebreak␣` within the words. like: manually-hyphenated.

2. using the command `\hyphenation{word1 word2 ...}` in the preamble to specify how to hyphenate certain words. like: `\hyphenation{man-u-script}`, manuscript manuscript manuscript, you can see the effect here. If the **inputenc** package is used with an appropriate encoding option, such as **utf8**, **latin1**, etc., the hyphenation of words is done automatically.

3 Document Layout and Organization

3.1 Document class

The syntax of the document class is:

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

Standard documents classes include: *article, report, book, letter, slides, poster, handout, and memoir.*

3.1.1 Standard class options

The **options** allow one to customize the document layout and behavior. Common options are as follows:

Font size Usually, the font size options are **10pt, 11pt, 12pt**, one can also use other sizes as well.

Paper size

- *a3paper* (297mm × 420mm)
- *a4paper* (210mm × 297mm)
- *a5paper* (148mm × 210mm)
- *b5paper* (182mm × 250mm)
- *letterpaper* (216mm × 279mm)
- *legalpaper* (216mm × 356mm)
- *executivepaper* (184mm × 267mm)

and *landscape* (contrast to *portrait* (default)) for landscape orientation, which rotate the page by 90 degrees and let the longer dimension be the horizontal one.

Page formats

- The text one the page may be farmatted into one or two columns with options *twocolumn* and *onecolumn* (one can also use package *multicol* flexibly).
- Different page numbering place may be specified with options *twoside* (default for *book*, page number are always on the outside) and *oneside* (usually default), respectively.

- Class *book* with *openright* (default, the chapters start on a right-hand, odd-numbered page) or *openany* (a chapter always starts on the nest page).
- *titlepage* (default for *report and book*, the title page is separated from the main content) or *notitlepage* (default for *article* and *ctexart*), respectively.

Further options

- *leqno*: number equations on the left-hand side instesd of the normal right-hand side.
- *fleqn*: display equations with left-hand side indentation instead of centered.
- *openbib*: enlarge the space between the bibliography, but *using package* ***biblatex*** *is recommended.*
- *draft*: display a draft version, which can emphasize wrong parts in the document.
- *final*: display the final version, which remove all draft options.

Some options have its own parameters, for *fleqn*, one can use `\set{\mathindent}{2cm}` to set the indented length of the equations. And `\columnsep` can be used to set the space between two columns, `\columnseprule` can be used to set the cut-off rule width between two columns, default 0pt(without dividing line).

3.1.2 Loading packages

One can use `\usepackage` to load packages (namely .sty files). The syntax is:

```
\usepackage[opt1, opt2, ...]{packagename1, packagename2, ...}[yyyy/mm/dd]
```

If several packages take the same option, it's only necessary to declare it once in **\documentclass**, because the options are passed to all loaded packages globally. By contrast, any options specified with `\usepackage` will only apply to the loaded packages listed in that one command.

One may load a package with a specific version. e.x., `\usepackage{package}[2023/05/23]` will load the *inputenc* package with version 2023/05/23.

3.2 Page style

Its form is:

```
\pagestyle{style}
```

The mandatory `\pagestyle` command must be placed after `\documentclass`, but before any other commands that affect the page style. The argument `style` can takes one of the following values:

- *plain*: the page number is in the lower right corner or lower center (depends on the document class), and the page title is not displayed (page head is empty).
- *headings*: the footer is empty, and the page title is displayed.
- *myheadings*: the same as *headings*, but the page title is chosen manually by `\markboth` or `\markright` command. ex., `\markboth{left}{right}` will display *left* on the left-hand side of the footer, and *right* on the right-hand side.
- *empty*: no page number and title is displayed.

The command `\thispagestyle{style}` functions the same as `\pagestyle`, but only apply to the current page.

3.2.1 Heading declarations

For *twoside* document class (even-numbered pages are considered as left-hand pages), the declaration `\markboth{left head}{right head}` will display *right head* on the right-hand or even-numbered page side (the same as `\markright{right head}`), and *left head* as page title on the left-hand or odd-numbered page side.

If the document class is *oneside* (all pages are considered to be right-handed), In this case, the declaration `\markright{}` is appropriate.

In the most cases, the proper *template* you chose will help you to set the headings automatically.

3.2.2 Customized head and footlines

The package **fancyhdr** provides a flexible way to customize the head and footlines ("hdr" means header and footer). One can use these command to customize the head and footlines:

```
\lhead{left head}  \chead{center head}  \rhead{right head}
\lfoot{left foot}  \cfoot{center foot}  \rfoot{right foot}
```

or, generally recommended,

```
\fancyhead[L]{left head}  \fancyhead[C]{center head}  \fancyhead[R]{right head}
\fancyfoot[L]{left foot}  \fancyfoot[C]{center foot}  \fancyfoot[R]{right foot}
```

or, for *twoside* document class,

```
\fancyhead[LE,RO]{left head}  \fancyhead[CE,RO]{center head}  \fancyhead[RE,LO]{right head}
\fancyfoot[LE,RO]{left foot}  \fancyfoot[CE,RO]{center foot}  \fancyfoot[RE,LO]{right foot}
```

where 'LE,RO' means Left part on Even-numbered pages and Right part on Odd-numbered pages.

Rule will decorate the head and footlines defaultly, which is useful if you write a less important paper. But remember to use `\pagestyle{fancy}` after claim one of the above options in the preamble, otherwise the head and foot will not be displayed.

To change the width of the rules, one can use:

```
\renewcommand{\headrulewidth}{1pt}
\renewcommand{\footrulewidth}{1pt}
```

3.2.3 Page numbering

To specify the style of page numbering, one can use:

```
\pagenumbering{style}
```

where *style* can be one of the following:

- *arabic*: Arabic numerals 1, 2, 3, ..., default style.
- *roman*: lowercase roman numerals i, ii, iii, ...
- *Roman*: uppercase roman numerals I, II, III, ...
- *alph*: lowercase letters a, b, c, ...
- *Alph*: uppercase letters A, B, C, ...

To reset the page number to what you want, one can use:

```
\setcounter{page}{number} % note: 'page' is not an option here.
```

then the *number* will appear on current page and count forward.

3.2.4 Paragraph formatting

Use `\setlength{options}{value}`, options for paragraph formatting can be:

- `\parindent`: the indentation of the first line of a paragraph
- `\parskip`: the vertical space between paragraphs
- `\baselineskip`: the minimum space between the bottom of two lines (baseline)
- `\lineskip`: the vertical space between lines

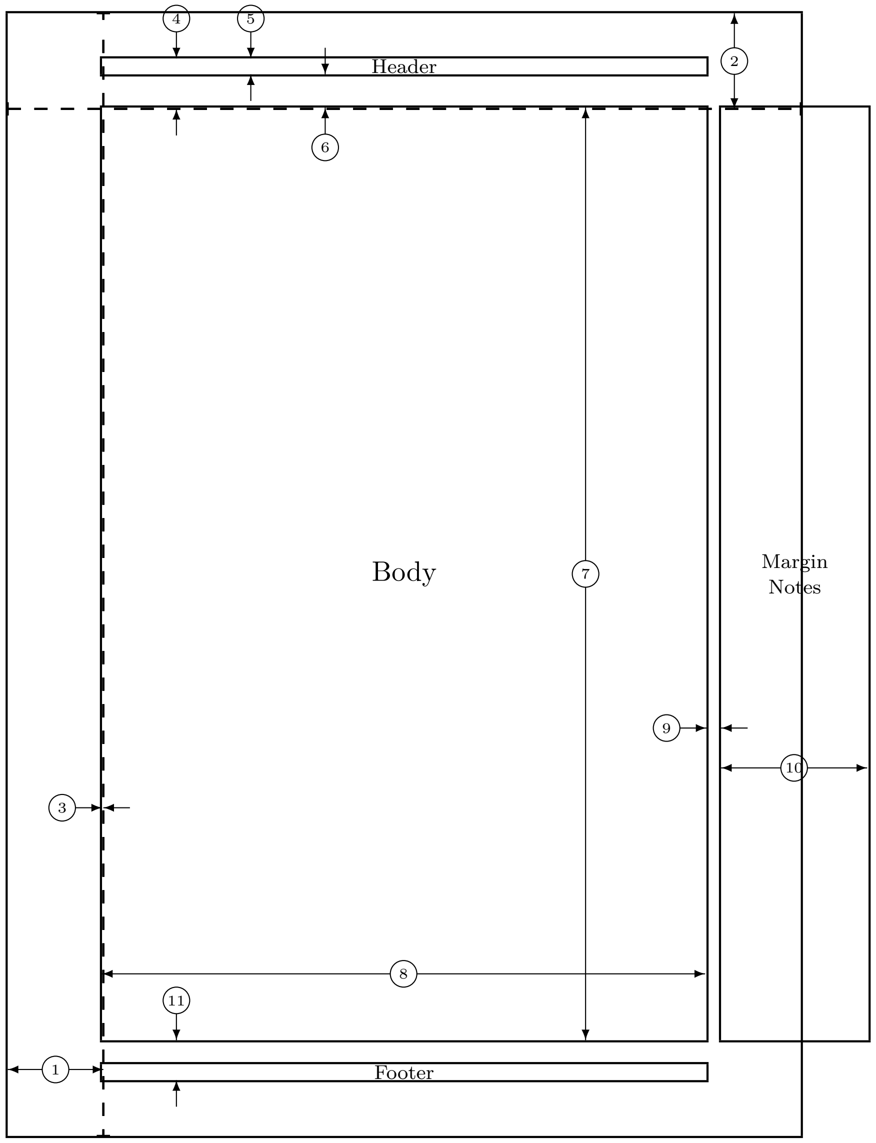
For `\baselineskip`, beside `\setlength`, the command `\renewcommand{\baselineskip}{factor}` can change the baseline skip factor as well. ex., `\renewcommand{\baselineskip}{1.2}` will make the baseline skip 20% larger than the default value.

To suppress or force the indentation of paragraph:

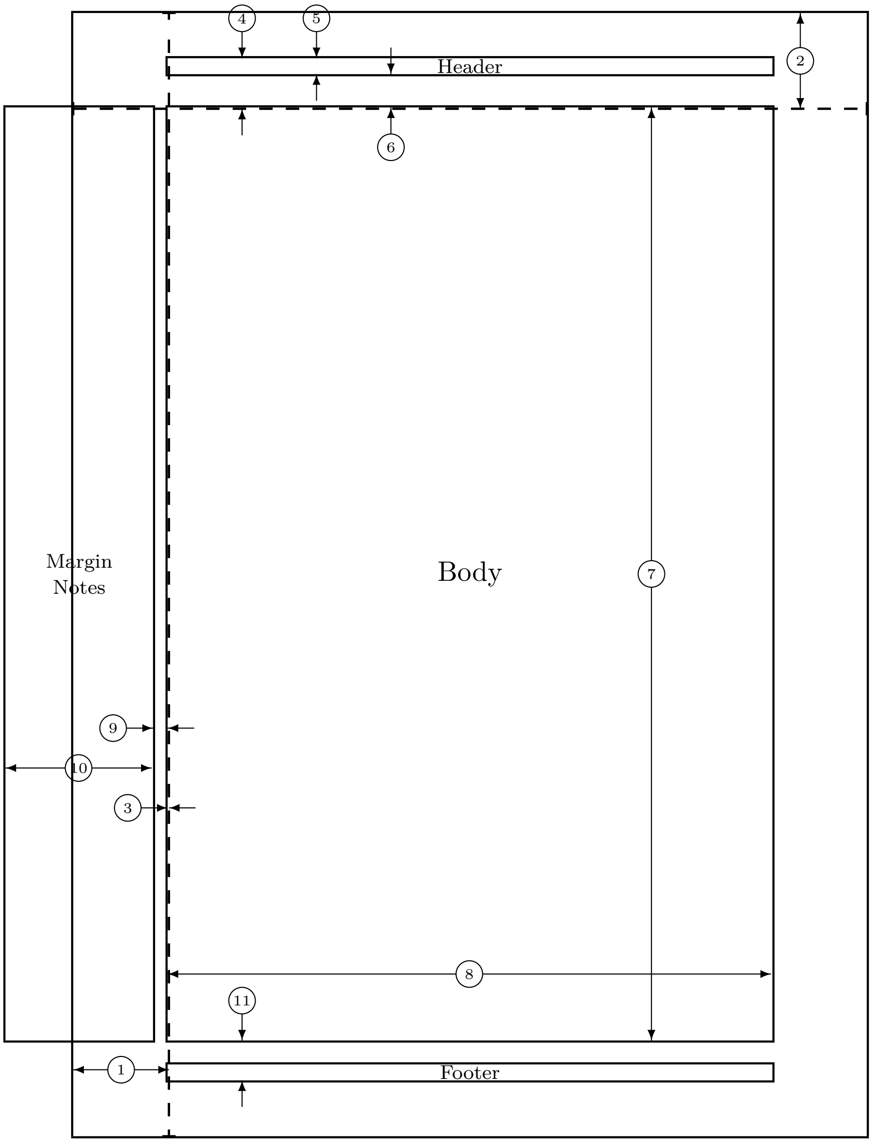
```
\noindent   or   \indent
```

3.2.5 Page layout

Use package **layout** and command `\layout` to display the current page layout parameters, Which is useful for debugging. Like this:



1	`one inch + \hoffset`	2	`one inch + \voffset`
3	`\oddsidemargin = -1pt`	4	`\topmargin = -38pt`
5	`\headheight = 12pt`	6	`\headsep = 25pt`
7	`\textheight = 702pt`	8	`\textwidth = 455pt`
9	`\marginparsep = 11pt`	10	`\marginparwidth = 111pt`
11	`\footskip = 30pt`		`\marginparpush = 5pt (not shown)`
	`\hoffset = 0pt`		`\voffset = 0pt`
	`\paperwidth = 597pt`		`\paperheight = 845pt`



1	one inch + \hoffset	2	one inch + \voffset
3	\evensidemargin = -1pt	4	\topmargin = -38pt
5	\headheight = 12pt	6	\headsep = 25pt
7	\textheight = 702pt	8	\textwidth = 455pt
9	\marginparsep = 11pt	10	\marginparwidth = 111pt
11	\footskip = 30pt		\marginparpush = 5pt (not shown)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 597pt		\paperheight = 845pt

The layout parameter showed above can set by using `\setlength{parameter}{value}`. However, it's tedious to set all the parameters one by one, so package **geometry** is recommended to set the page layout easily. In the preamble of this document, I have used:

```
\usepackage[a4paper, left=2.5cm, right=2.5cm, top=2.5cm, bottom=2.5cm]{geometry}
```

or, same as:

```
\usepackage[a4paper, margin=2.5cm]{geometry}
```

or, split the options from `\usepackage`:

```
\usepackage{geometry}
\geometry{a4paper, left=2.5cm, right=2.5cm, top=2.5cm, bottom=2.5cm}
```

Of course, there are many other options provided by package **geometry**, like:

- *hmargin* or *left right*: set the left and right margins explicitly.
  If you set hmargin=1in, the left margin and the right margin will both be 1in.
  If you set left=1in, the left margin will be 1in, and one can continue set the right margin.
- *vmargin* or *top bottom*: set the top and bottom margins explicitly. The same as *hmargin*.
- *heightrounded*: make the text height an integer number of lines
- *showframe*: show the layout frame for debugging
- *scale*: scale the text area to a certain factor of the paper size, default `scale={0.7, 0.7}` or `hscale=0.7, vscale=0.7` or `scale=0.7`, namely 70% of the paper size.
- *etc.* (Please refer to the package documentation for a complete list of settings.)

3.2.6 Multiple columns

As saying in 2.6.4, it's recommended to use package **multicol** to typeset multiple columns. For example:

```
\begin{multicols}{2}[header text][0.5pt]
    ...
\end{multicols}
```

There are some lengths to control the multiple columns layout, like:

- `\columnsep`: the horizontal space between columns
- `\columnseprule`: the width of the rule between columns, it's a visible line with none zero width.

- `\columnseprulecolor`: the color of the rule between columns
- `\premuticols`: the vertical space before the first column.
  If the remaining space is not enough (less than `\premuticols`), the first column will be typeset in the next page.
  This '0.5pt' in `\begin{multicols}{2}[header text][0.5pt]` also specify the `\premuticols` value.
- `\postmulticols`: the vertical space after the last column

3.3 Parts of the document

3.3.1 Title page

A title page can be created with the commands:

```
\title{title}
\author{author}
\date{date}
\maketitle
```

The information part

```
\title{title}
\author{author}
\date{date}
```

can be placed in the preamble or before `\begin{document}`.

If there are several authors, one can use `\and` to separate them. Like:

```
\author{Author1\\Institute1\\Address1 \and Author2\\Institute2\\Address2}
```

If `\date` is omitted, the current date will be used automatically.

The command `\thanks{Footnote}` can be given after the author name, title, or date to add a footnote to the title page.

If you desire to customize the title page, you can use package **titling** or define *titlepage* environment.

3.3.2 Abstract

An abstract can be created with the `abstract` environment:

```
\begin{abstract}
```

```
    \textbf{Abstract:} This is the abstract text...
    \textbf{Keywords:} keyword1; keyword2; keyword3.
\end{abstract}
```

3.3.3 Sections and chapters

The commands ordered by priority for sections and chapters are:

```
\part{part title}          % only for 'book' and 'report' class
\chapter{chapter title}                  % only for 'book' and 'report' class
\section{...}
\subsection{...}
\subsubsection{...}
\paragraph{...}
\subparagraph{...}
```

The syntax of these commands is the same, like:

```
\sec_command[short title]{full title}   or   \sec_command*{star-form title}
```

where *full title* will appear in the document, and *short title* will appear in the table of contents and page headings.

For *starred form*, the section or chapter will not be numbered, and will not appear in the table of contents and page headings.

If you intend to refer to a section or chapter later, use:

```
\label{sec:label}   % after the section or chapter command
...
As mentioned in Section~\ref{sec:label}, ...
```

or use `\pageref{}` like:

```
\label{sec:label}   % after the section or chapter command
...
As mentioned on page \pageref{sec:label}, ...
```

to refer to the page number where the section or chapter appears.

If you process several TeX files separately and want to combine them into a single pdf, you may want to reset the chapter number counter for each file. To achieve this, use:

```
\setcounter{chapter}{0}   % reset chapter number to 0
```

However, `\ref` only gives the counter value of the section or chapter. So if you want to refer to the section or chapter with its name, you can use package **nameref** and command `\nameref` like:

```
\label{sec:label}   % after the section or chapter command
...
As mentioned in Section~\nameref{sec:label}, ...
```

or use package **hyperref** and command `\hyperref` like:

```
\label{sec:label}   % after the section or chapter command
...
As mentioned in Section~\hyperref[sec:label]{any name you want}, ...
```

Note that `nameref` is part of package **hyperref**, so you can use `\nameref` with package **hyperref** as well.

3.3.4 Appendix

An appendix is introduced with the declaration:

```
\appendix
```

and the chapter/section number will be reset and change the form from numerals to letters A, B, C, etc. And the word *Chapter* will be replaced by *Appendix*.

3.3.5 Book structure

In class **book**, the structure can be simplified by the commands:

```
\frontmatter % for preface, table of contents, etc.
\mainmatter % for main body of the book
\backmatter % for appendices, index, references/bibliography, colophon, etc.
```

here the `\frontmatter` command switches page numbering to Roman numerals and suppresses the numbering of chapters; `\mainmatter` resets the page numbering to 1 with Arabic numbers and reactivates the chapter numbering; this is once again turned off with `\backmatter`.

3.4 Table of contents and Lists

3.4.1 Print TOC

To print a table of contents(TOC), use:

```
\tableofcontents
```

The sectioning depth to which entries(TOC items) are made in the TOC can be set in the preamble:

```
\setcounter{tocdepth}{2}
```

the command above sets TOC depth to 2 (that is, sections and subsections).

If you use `\tableofcontents` in the document, you will find a `.toc` file generated in the same directory as the `.tex` file. Because for the information in the table of contents is to be printed near the beginning of the document, information that cannot be known until the end.

LaTeX solves this problem as follows: The first time the document is processed, no table of contents can be included, but instead LaTeX opens a new file with the same name as the source file but with the extension `.toc` ; the entries for the table of contents are written to this file during the rest of the processing.

3.4.2 Add entries to TOC

Note that the *-form of sectioning commands, like `\section*{...}`, do not add entries to the TOC automatically. To insert them manually, use:

```
   \addcontentsline{toc}{section}{entry text}
or \addtocontents{toc}{entry text}
```

where `{section}` can be replaced by `{subsection}`, etc. and `{entry text}` is the text to be displayed in the TOC. for example:

```
\section*{this is a section}
\addcontentsline{toc}{section}{this is a section}
```

The command `\addtocontents{toc}{entry text}` can puts any desired command or text into the `.toc` file.

3.4.3 List other items

Beside TOC, the list of figures and tables can also be generated with:

```
\listoffigures % create .lof file
\listoftables  % create .lot file
```

the entries in the list of figures and tables are added automatically when the `\caption` command in the *figure* or *table* environment is used.

Analogously, one can add entries to the list of figures and tables manually with:

```
\addcontentsline{lot}{table}{entry text}
\addtocontents{lot}{entry text}
\addcontentsline{lof}{figure}{entry text}
\addtocontents{lof}{entry text}
```

4 Text Display

4.1 Font style

4.1.1 Emphsis

Italics is used to *emphasize* a word or a phrase. Use the *declaration*:

```
\em
```

or the *command*:

```
\emph{text}
```

or the environment:

```
\begin{em}
    text
\end{em}
```

to make the text appears in italics.

Nested *emphsis* is supported, for example,

```
\emph{first, \emph{second, and \emph{third font switch.}}}
```

first, second, and *third font switch.*

4.1.2 Font size

The following declarations are available in LaTeX for changing the font size:

`\tiny`	smallest
`\scriptsize`	very small
`\footnotesize`	smaller
`\small`	small
`\normalsize`	normal
`\large`	large
`\Large`	larger
`\LARGE`	even larger
`\huge`	still larger
`\Huge`	largest

4.1.3 Line spacing

For every font size, there is a corresponding natural line spacing `\baselineskip` or `\linespread` / `\baselinestretch`. The `\baselineskip` is a length and the `\linespread` is a factor, for e.x.

```
    \setlength{\baselineskip}{10pt} % set line spacing to 10pt
    \linespread{1.5}                % 1.5 times the normal line spacing
or
    \renewcommand{\baselinestretch}{1.5}
```

The new value of `\baselinestretch` / `\linespread` does not take effect until the next change in fontsize. To implement a new value in the current font size, it is necessary to switch to another size and back again immediately. If the present font size is `\normalsize`, the sequence

```
    \small\normalsize
```

will do the trick. Because the true interline spacing is $\backslash baselinestretch \times \backslash baselineskip$, and the command `\renewcommand{\baselinestretch}{1.5}` only changes the factor, but not completes the calculation of the new interline spacing.

4.1.4 Font attributes

For normal usage there are some declarations and corresponding commands:

family: for the overall style.

\rmfamily	to switch to a Roman font
\sffamily	to switch to a Sans serif font
\ttfamily	to switch to a `Typewriter font`

Shape: for the form of the font.

\upshape	to switch to an upright shape
\itshape	to switch to an *italic shape*
\scshape	to switch to a SMALL CAPS SHAPE
\slshape	to switch to a *slanted shape*

Series: for the width or weight (boldness) of the font.

\mdseries	to switch to a medium weight
\bfseries	to switch to a **boldface weight**

These declarations are used just like any others, normally enclosed in a pair of curly braces { ... }, such as `{\scshape S.C. shape example}` to produce S.C. SHAPE EXAMPLE. Or use an environment form:

```
\begin{font_style}
    text
\end{font_style}
```

for longer sections of text.

More non-standard font attributes, see section A.1 in the textbook.

Finally, the declaration `\normalfont` resets all the attributes (except size) back to their defaults: Roman, upright, medium weight.

4.1.5 Font commands

For each of the above font attributes, there is a corresponding command that takes an argument *text* and applies the attribute to that text only.

Family:	\textrm{text}	\textsf{text}	\texttt{`text`}	
Shape:	\textup{text}	\textit{*text*}	\textsc{TEXT}	\textsl{*text*}
Series:	\textmd{text}	\textbf{**text**}		
default:	\textnormal{text}			
Emphsis:	\emph{*text*}			

4.1.6 Additional fonts

It is likely that your computing center or your TEX installation has even more fonts and sizes than those listed above. To load a new font explicitly by name, the command

```
     \newfont{\fnt}{name scaled factor}
or
     \newfont{\fnt}{name at size}
```

is given, where *name* is the font name, *factor* is a scaling factor (e.x. 1200 for 120% size), and *size* is the desired size in points.

For example, to install a slanted, sans serif font of size 20.74 pt, as `\sss`, use the command

```
\newfont{\sss}{cmssi17 at 20.74pt}
```

Now the declaration `\sss` can be used to switch to that font.[1]

[1] “cmssi17” means Computer Modern Sans Serif Italic at 17 pt

4.2 Centering and indenting

4.2.1 Centered text

Use the environment

```
\begin{center}
    line 1      \\[len]
    line 2      \\[len]
    line 3
\end{center}
```

or the declarations

```
\centering
```

Center a single line, use

```
\centerline{line}
```

4.2.2 One-sided justification

Use the environments

```
\begin{flushleft}                   \begin{flushleft}
    line 1  \\[len]                     line 1  \\[len]
    line 2  \\[len]         or          line 2  \\[len]
    line 3                              line 3
\end{flushleft}                     \end{flushleft}
```

or the declarations

```
\flushleft  or  \flushright
```

4.2.3 Two-sided indentation: quotations

A section of text may be prominently displayed by indenting it by an equal amount on both sides, with the environments

```
\begin{quote} line 1 \\[len]  line 2 \\[len] ... line n \end{quote}
\begin{quotation} line 1 \\[len]  line 2 \\[len] ... line n \end{quotation}
```

In the **quotation** environment, paragraphs are marked by extra indentation of the first line, whereas in the **quote** environment, they are indicated with more vertical spacing between them.

4.2.4 Verse indentations

For indenting rhymes, poetry, verses, and so forth on both sides, the environment

```
\begin{verse} poem \end{verse}
```

is more appropriate.

The above indenting schemes may be nested inside one another. A maximum of six such nestings is allowed.

4.3 Lists

4.3.1 Itemsize and enumerate

Use the environment *env_name* (*itemize*, *enumerate*) as follows:

```
\begin{env_name}
    \item item 1
    \item item 2
    \item item 3
\end{env_name}
```

to produce an itemized list like:

- The individual entries are indicated with a black dot, known as a *bullet*, as the label.
- The text in the entries may be of any length. The label appears at the beginning of the first line of text.
- Successive entries are separated from one another by additional vertical spacing.

and an enumerate list like:

1. The labels consist of sequential numbers.
2. The numbering starts at 1 with every call to the `enumerate` environment.

4.3.2 Description

Description has a bit different usage from *itemize* and *enumerate*. Use the environment *description* as follows:

```
\begin{description}
    \item[term 1] description 1
    \item[term 2] description 2
    \item[term 3] description 3
\end{description}
```

to produce a description list like:

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

example A keyword is used as the label and the entry contains a clarification or explanation.

other uses It may also be used as an author list in a bibliography.

4.3.3 Nested lists

The above lists may be included within one another, either mixed or of one type, to a depth of **four** levels. The type of label used depends on the depth of the nesting. For example,

```
\begin{itemize}
    \item The \texttt{itemize} label at the first level is a ...
    \begin{enumerate}
        \item The numbering is with Arabic numerals since this ...
        \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 lowercase 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 \verb+\item+ command ...
\end{itemize}
```

generates:

- The `itemize` label at the first level is a ...
  1. The numbering is with Arabic numerals since this ...
     – 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 lowercase letters ...

    – The label at this level is a long dash.

  2. Every list should contain at least two points.

- Blank lines ahead of an `\item` command ...

4.3.4 Changing label style

Like *description*, the labels, or markers, used in the itemize and enumerate environments can be easily changed by means of the optional argument in the `\item` command. With `\item[+]` the label becomes +, and with `\item[2.1:]` it is 2.1:.

It is also possible to change the standard labels for all or part of the document. The labels are generated with the internal commands:

```
\labelitemi , \labelitemii , \labelitemiii , \labelitemiv
\labelenumi , \labelenumii , \labelenumiii , \labelenumiv
```

The endings `i`, `ii`, `iii`, and `iv` refer to the four possible levels. These commands can be redefined with `\renewcommand`. For example, to change the label for the first level of **itemize** from bullet (*) to +, use `\renewcommand{\labelitemi}{+}`.

There is a counter for each **enumerate** level, named `enumi`, `enumii`, `enumiii` and `enumv`. The current value of a counter can be printed using one of the commands

```
\arabic{counter_name}   \roman{counter_name}   \Roman{counter_name}
\alph{counter_name}     \Alph{counter_name}
```

They can be used in the redefinitions of the label commands. For example, to change the second-level label to Arabic number followed by '.)', it's necessary to give:

```
\renewcommand{\labelenumii}{\arabic{enumii}.)}
```

or use more than one counter as one desires:

```
\renewcommand{\labelenumii}{\Alph{enumi}.\arabic{enumii}}
```

which will generate the form A.1, A.2 ... B.1, B.2 ... and so on.

4.4 Generalized lists

4.5 Theorem-like declaration

At first, we present an example:

```
\newtheorem{axiom}{Axiom}[subsection]
```

```
\begin{axiom}[extra title]
The natural numbers form a set S of distinct elements. For any two
elements a, b, they are either identical, a = b,
or different from one another, a $\neq$ b.
\end{axiom}
```

which generates:

Axiom 4.5.1 (extra title)**.** *The natural numbers form a set S of distinct elements. For any two elements a, b, they are either identical, $a = b$, or different from one another, $a \neq b$.*

The operation is divided into two steps. firstly, we define the structure type and its title:

```
\newtheorem{struct_type}{struct_title}[in_counter]
```

Here *struct_type* is the user's arbitrary designation for the structure, while *struct_title* is the word that is printed in boldface followed by the running number (for example, **Theorem**).

If the optional argument *in_counter* is missing, the numbering is carried out sequentially throughout the entire document. While if the name of an existing counter, such as **subsection**, is given for *in_counter*, the numbering is reset every time that counter is augmented, and both are printed together, as in **Axiom 4.5.1** above.

Secondly, use the predifined structures as environments:

```
\begin{struct_type}[extra_title] text \end{struct_type}
```

which also increments the necessary counter and generates the proper number.

Occasionally a structure is not numbered on its own but together with another structure. This can be included in the definition with another optional argument:

```
\newtheorem{struct_type}[num_like]{struct_name}
```

where *num_like* is the name of an existing theorem structure that shares the same counter. Thus by defining `\newtheorem{subthrm}[theorem]{Sub-Theorem}`, the two structures theorem and subthrm will be numbered as a single series: **Theorem 1**, **Sub-Theorem 2**, **Sub-Theorem 3**, **Theorem 4**, and so on.

4.6 Printing Literal text

4.6.1 Verbatim

To print text exactly as it is typed, use the environment:

```
\begin{verbatim} text \end{verbatim}
\begin{verbatim*} text \end{verbatim*}
```

With the *-form, blanks are printed with the symbol ␣ to make them visible.

For inline verbatim text, use the command `\verb`, and brace *text* with "=" or "|", like:

```
\verb|word of text|  \verb*|word of text|
\verb=word of text=  \verb*=word of text=
```

```
word of text   word␣of␣text

word of text   word␣of␣text
```

if the symbol '=' and '|' has appeared in the *text*, one should use others like '+', or any other characters as the delimiter.

Important: Neither the verbatim environment nor the `\verb` command may be used in an argument of any other command![2]

The **verbatim** package offers two other extra features. It provides a **comment** environment that simply ignores its contents, as though each line started with a % sign. And it adds a command `\verbatiminput{filename}` to input the specified file as literal text. This is useful for listing actual computer programs rather than copying them into the LaTeX file.

4.6.2 Alltt and shortvrb

The standard package **alltt** provides an **alltt** environment that also prints its contents literally in a typewriter font, except that the characters `\{}` retain their normal meaning. Thus LaTeX commands can be included within the literal text. For example,

```
\begin{alltt}
Underlining \underline{typewriter}
text is also possible.
Note that dollar ($) and
percent (%) signs are
treated \emph{literally}.
\end{alltt}
```

```
Underlining typewriter
text is also possible.
Note that dollar ($) and
percent (%) signs are
treated literally.
```

The standard package **shortvrb** offers a shorthand for the `\verb` command. After issuing `\MakeShortVerb{\|}` (let '|' be the verbatim switch), one can print short literal text with |*text*|. The countering command `\DeleteShortVerb{\|}` then restores the original meaning to |. Any character may be temporarily turned into a literal switch this way.

4.6.3 Listings

To print code listings, use the environment **lstlisting** from the package **listings**. For example,

[2] To avoid any errors may be caused in this case, use package **cprotect** and its command `\cprotect` before the command whose argument contains verbatim text, like `\cprotect\footnote{\verb|text|}`.

```
\begin{lstlisting}[style=C]
int main() {
    printf("Hello, World!\n");
    return 0;
}
\end{lstlisting}
```

```
int main() {
    printf("Hello, World!\n"
       );
    return 0;
}
```

Before using *style=C*, one should define it with `\lstdefinestyle{C}{...}` in the preamble. I have defined *style=default* and *style=C* (inherit from *style=default*) as follows in the document:

```
\lstdefinestyle{default}{                   % default style
backgroundcolor=\color{backcolour},
commentstyle=\color{codegreen},
keywordstyle=\color{blue}\bfseries,
numberstyle=\tiny\color{codegray},
stringstyle=\color{codepurple},
basicstyle=\ttfamily\normalsize,
breaklines=true,                            % auto line break
showstringspaces=false,                     % Do not show spaces in strings
captionpos=b,                               % Caption position: bottom
mathescape=true,                            % Allow math mode in listings
}

\lstdefinestyle{C}{
style=default,                        % Inherit all settings of the default style
language=C,                                 % Specify the language as C
keywordstyle=\color{darkblue}\bfseries, % Override: keywords become dark blue
stringstyle=\color{red},                    % Override: strings become red
}
```

where the command `\color{}` is from package **xcolor**.

4.6.4 Email and internet addresses

E-mail and Internet addresses present some special problems that are solved with the **url** package by Donald Arseneau. One can opt for a more advanced approach by using the **hyperref** package. The syntaxes are

```
\url{url}
\href{url}{display text}
```

Inserting the address with the `\url` command allows it to be broken at non-letters between words, without a hyphen. For example,

```
An Internet address may be given as
\url{http://address.edu/home/page/},
or \href{https://www.google.com}{google}.
An e-mail address might be given as
\url{fred.smith@general.services.gov}
or \href{example@example.com}{email}.
```

An Internet address may be given as http://address.edu/home/page/, or google. An e-mail address might be given as fred.smith@general.services.gov or email.

A fixed address can be predefined with, for example,

```
\urldef{\myurl}
\url{myname@mydomain.cn}
```

Now `\myurl` is used to print myname@mydomain.cn.

4.7 Comments within text

In LaTeX, the comment character is the percent sign %.

For more than sigle line, use the **comment** environment (4.6.1).

5 Boxed Text

LaTeX offers the user a choice of three box types: LR boxes, paragraph boxes, and rule boxes. The LR (left-right) box contains material that is ordered horizontally from left to right in a single line. A paragraph box will have its contents made into vertically stacked lines of a given width. A rule box is a rectangle filled solidly with black, usually for drawing horizontal and vertical lines.

In this chapter, I will give the **original code** and the **display** of each box type because of the increasing complexity of verbal expressions.

5.1 LR boxes

5.1.1 Basic usage

To create an LR box containing text in a single line, use:

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

For example,

```
\mbox{text1} and \makebox[2cm][c]{text2}, \fbox{text3} and \framebox[2cm][c]{text4}
```

text1 and   text2   , $\boxed{\text{text3}}$ and $\boxed{\quad\text{text4}\quad}$

one can find that the `\mbox` and `\makebox` commands produce the same result but latter one allows one to specify the width and position of the box. While the `\fbox` and `\framebox` commands produce a visible frame around the text.

The parameters *pos* can be:

```
l  % to left justify the text in the box
c  % to center justify it
r  % to right justify it
s  % to stretch the text to fill the box width
```

one can skillfully justify the text in the box, like `\makebox[0pt][l]{/}S` produce $\not{S}$, which overlap the slash on the S.

The unit of *width* can chose basic LaTeX length like `pt` (point), but it's also possible to use its natural dimensions:

```
\width          % is the natural width of the box
\height         % is the distance from baseline to top
\depth          % is the distance from baseline to bottom
\totalheight    % is the height of the box, equal to \height + \depth
```

5.1.2 Storing and recalling a LR box

If a set piece of text is to appear in several places within the document, it can be stored by first, and then recall by `\usebox{name}`.

Original code:

```
\newsavebox{\boxname}
        % create a LR box named boxname, \ is necessary
\sbox{\boxname}{text in box}
        % store the text in boxname
% \savebox{\boxname}[1pt][c]{text}
        % another saving method, with 1pt offset and center justify
I use \usebox{\boxname} at first. I use \usebox{\boxname} again.
```

Display:

I use text in box to display at first. I use text in box again.

The stored contents are inserted into the document text wherever desired, as a single unit. The contents of an LR box may also be stored with the environment:

```
\begin{lrbox}{boxname}
text
\end{lrbox}
```

it's the same as `\sbox{\boxname}{text}`.

vertical shifting The command

```
\raisebox{lift}[height][depth]{text}
```

produces an `\mbox` with contents *text*, raised above the current baseline by an amount *lift*. The *height* and *depth* parameters are optional. Without these arguments, the box has its natural size determined by *text* and *lift*.

For example,:

```
Baseline \raisebox{1ex}{high} and \raisebox{-1ex}{low} and back again
```

Baseline high and low and back again

5.1.3 Box style parameters

There are two style parameters for the frame boxes `\fbox` and `\framebox` that may be reset by the user:

```
\fboxrule  % determines the thickness of the frame lines,
\fboxsep   % sets the amount of spacing between the frame and enclosed text.
```

New values are assigned to these length parameters in the usual LaTeX manner with the command `\setlength`.

5.2 Paragraph boxes

5.2.1 Parboxes and minipages

Whole paragraph can be put into separate *parboxes* by the command:

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

or with the `\minipage` env.

```
\begin{minipage}[pos]{width}
 your text
\end{minipage}
```

the argument *pos* can take on the values:

```
b   % align the bottom edge of the box with the current baseline
c   % default and center justify
t   % align the top line of text with the current baseline
```

Two examples:

```
\parbox[b]{3.5cm}{\sloppy This is a 3.5 cm wide parbox. It is
vertically bottom justify on the}
\hfill CURRENT LINE \hfill
\parbox[t]{5.5cm}{Narrow pages are hard to format. They usually
produce many warning messages on the monitor. The command
\texttt{\symbol{92}sloppy} can stop this.}
```

This is a 3.5 cm wide parbox. It is vertically bottom justify on the

CURRENT LINE

Narrow pages are hard to format. They usually produce many warning messages on the monitor. The command `\sloppy` can stop this.

```
\begin{minipage}[b]{4.3cm}
The minipage environment creates a vertical box like the parbox
```

```
command. The bottom line of this minipage is aligned with the
\end{minipage}\hfill
\parbox{3.0cm}{middle of this narrow parbox, which in turn is
aligned with}
\hfill
\begin{minipage}[t]{3.8cm}
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 isaligned 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.

5.2.2 Parbox with specific height

The complete syntax of the `\parbox` command and minipage environment includes two more optional arguments *height* and *inner_pos*:

```
    \parbox[pos][height][inner_pos]{width}{text}
or
    \begin{minipage}[pos][height][inner_pos]{width}
     your text
    \end{minipage}
```

The argument *height* is a length specifying the height of the box, it's unit can be set the same as *width* of `\makebox` and `\framebox` (use the basic unit of length like `pt`, or `\width`, `\height`, `\depth`, `\totalheight`).

The optional argument *inner_pos* states how the text to be positioned internally. 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, default
s  % to stretch it to fill up the whole box.
```

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.

There's an example which shows the role of *inner_pos*:

```
\begin{minipage}[t][2cm][t]{3cm}
This is a minipage of height 2cm with the text
at the top.
\end{minipage}\hrulefill
\parbox[t][2cm][c]{3cm}{In this parbox, the text
is centered on the same height.}\hrulefill
\begin{minipage}[t][2cm][b]{3cm}
In this third paragraph box, the text is at the bottom.
\end{minipage}
```

This is a minipage of height 2cm with the text at the top.

In this parbox, the text is centered on the same height.

In this third paragraph box, the text is at the bottom.

5.3 Rule boxes

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

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

where *lift* is an optional length to raise the box above the current baseline. The parameters *lift*, *width* and *height* are all lengths.

However, rule boxes with zero width or height are often used to create horizontal or vertical lines (so-called a **strut**). For example, to produce $\boxed{\text{Text}}$ rather than $\boxed{\text{Text}}$, you can use `\rule[-2mm]{0cm}{6mm}` to create "an invisible bar beginning 2mm below the baseline and 6mm long" inside the command `\fbox{}`, namely `\fbox{\rule[-2mm]{0cm}{6mm}Text}`.

5.4 Nested boxes

A parbox inside an `\fbox` command has the effect that the entire parbox is framed. The present structure was made with

```
\fbox{\fbox{\parbox{10cm}{A parbox...}}}
```

This is a parbox of width 10 cm inside a framebox inside a second framebox, which thus produces the double-framing effect.

5.5 Further framed boxes

fancybox

The **fancybox** package, by Timothy van Zandt, allows additional framed boxes of various styles. These new framed boxes are:

\shadowbox{text}
The width of the shadow is given by the length **\shadowsize**, default 4pt. Multiline text must be placed in a minipage environment, the same as for **\fbox**.

\doublebox{text}
The default rule width of the inner frame is 0.75**\fboxrule**, that of the outer frame is 1.5**\fboxrule**, and the spacing between the frames is 1.5**\fboxrule** plus 0.5 pt.

\ovalbox{text}
The thickness of the frame is that of **\thinlines** (see textbook: Section 16.1.4); the diameter of the corners is set with the command **\cornersize{frac}**, to *frac* times the smaller of the box width or height, or with **\cornersize*{size}** to the length *size*. The default is *frac* =.5.

Also, the **fancybox** package provides the **\fancypage** command to create a framed box that fills current page. For e.x.:

```
    \fancypage{\setlength{\fboxsep}{5pt}
               \setlength{\shadowsize}{3pt}\shadowbox}{}
or simply:
    \fancypage{\shadowbox}{}
```

5.6 Footnotes and marginal notes

5.6.1 Standard footnotes

Footnotes are generated with the command:

```
\footnote{footnote_text}
```

The standard footnote marker is a small, raised number.[1]

The **\footnote** command may only be given within the normal paragraph mode, and not within math or LR modes. However, it may be used within a **minipage** environment, in which case the footnote text is printed beneath the minipage (5.6.4) and not at the bottom of the actual page.

[1] Normal American practice is to place the footnote number after any punctuation, whereas in Europe it is more usual to place it immediately after the word, preceding any punctuation mark.

5.6.2 Non-standard footnotes

To reset the footnote number to 1:

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

Since LaTeX performs the "+1" operation before displaying footnotes, setting it to 0 will cause the next footnote to be displayed as 1.

To implement a different style of footnote markers (default is arabic numerals), use the command:

```
\renewcommand{\thefootnote}{\number_style{footnote}}
```

where *number_style* is one of the counter commands including:

`\arabic`, `\roman`, `\Roman`, `\alph`, `\Alph` and `\fnsymbol`.

The last one `\fnsymbol` prints the counter values 1 through 9 as one of nine symbols: ∗(星号), †(单剑号), ‡(双剑号), §(分节号), ¶(段落号), ‖(双并行线), ∗∗, ††, ‡‡.

To specify the number of a footnote manually, use the optional argument *num*:

```
\footnote[num]{footnote_text}
```

5.6.3 Footnotes in forbidden modes

When using the **footnote** command whithin **math mode, tables or LR boxes**, the footnote itself is not generated. One need to split the footnote command into two parts

```
\footnotemark[num]                   % to generate the footnote mark only
\footnotetext[num]{footnote_text}    % to generate the footnote text only
```

If you use the *num* optional argument in both commands, the counter will not be incremented automatically. In the meanwhile, The 3 commands will have different influnce on the footnote counter without using *num*:

```
\footnotemark                   % counter + 1
\footnotetext{footnote_text}    % counter + 0
\footnote{footnote_text}        % counter + 1
```

Because `\footnotemark` and `\footnotetext` are always used together, the counter will be incremented automatically only when using the former.

The auto-count has solved the problem that multiple `\footnotemark` may have the same footnote number. If there are multiple `\footnotemark` commands in a forbidden mode, all of the latter `\footnotetext` outside the forbidden mode will have the same footnote number of the last `\footnotemark`. So one need to use:

```
\addtocounter{footnote}{dif_num}
```

to drift the footnote counter with *dif_num*, and use

```
\stepcounter{footnote}
```

to increase the footnote counter by 1 (the same as `\addtocounter{footnote}{1}`).

For example, in a table environment:

```
\begin{tabular}{ll}
Item A\footnotemark & Description of Item A \\
Item B\footnotemark & Description of Item B \\
\end{tabular}
\addtocounter{footnote}{-1}            % for 2 footnotemarks, drift back 1
\footnotetext{Footnote for Item A}
\stepcounter{footnote}                 % move forward 1
\footnotetext{Footnote for Item B}
```

to get:

Item A[2]	Description of Item A
Item B[3]	Description of Item B

5.6.4 Footnotes in minipages

Footnote commands are allowed inside the **minipage** environment. However, the footnote appears underneath the minipage, not below the main page:

> Footnote commands within a minipage[a] have a different marker style. The footnote comes after the next `\end{minipage}` command.[b] Minipage footnotes have a counter separate from that of the main page, called *mpfootnote*, counting independently of footnote.
>
> [a] The marker is a raised lowercase letter.
>
> [b] Watch out for nested minipages.

5.6.5 Marginal notes

Notes in the page margin are produced with the command

```
\marginpar{note_text}
```

which puts the text *note_text* into the margin beginning at the level of the line where the command is given.

[2] Footnote for Item A

[3] Footnote for Item B

By default, marginal notes appear in the right-hand margin of the page or in the outer margin when the *twoside* option has been selected. 'Outer margin' means the right-hand margin on odd-numbered pages and the left-hand margin on even-numbered pages. One can use the syntax

```
\marginpar[left_note]{right_note}
```

to include two versions of marginal notes for the left & right margins, depending on which one is selected. For example, the arrow appears at the side of this paragraph is generated by ⟸

```
\marginpar[\hfill$\Longrightarrow$]{$\Longleftarrow$}
```

The `\hfill` command above makes the arrow right-aligned.

5.6.6 Style parameters for footnotes and marginal notes

footnote style There are two footnote style parameters that may be changed as needed, either in the preamble or locally within an environment:

```
\footnotesep  % sets the vertical space between footnotes,
              %  changed it with \setlength
\footnoterule % defines the width of the horizontal rule
              % that separates footnotes from the main text.
```

To change `\footnoterule`, use

```
\renewcommand{\footnoterule}{\rule{wth}{hght}\vspace{-hght}}
```

where `\vspace{-hght}` is used to remove the extra vertical space between the rule and the first footnote.

marginal note style The following style parameters may be changed to redefine how marginal notes appear:

```
\marginparwidth   % the width of the margin box
\marginparsep     % the horizontal space between the main text and the margin box
\marginparpush    % the minimum vertical space between two marginal notes
```

6 Tables

6.1 Tabulator stops

Type	Quality	Color	Price
Paper	med.	white	low
Leather	good	brown	high
Card	bad	gray	med.

```
\begin{tabbing}
Type\qquad\= Quality\quad\=
Color\quad\= Price\\[0.8ex]
Paper   \> med. \> white \> low \\
Leather \> good \> brown \> high \\
Card    \> bad  \> gray  \> med.
\end{tabbing}
```

6.2 Tabular

6.2.1 Constructing tables

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

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

The **array** environment can only be applied in mathematical mode (see 9.4.4 Arrays). These environments actually create a minipage. The meaning of the arguments is as follows:

Argument	Meaning	Values
pos	vertical position	t[1], b[2]
width[3]	total width of table	any length
cols	column formatting	l, c, r, p{wid}[4], *{n}{c}[5], |, ||[6], @{text}[7]
rows	table contents	text, &, \\, \hline, \hline\hline[8], `\cline{m-n}`[9], `\multicolumn{num}{col}{text}`[10], `\vline`[11]

6.2.2 Table style parameters

There are a number of style parameters used in generating tables, which LaTeX sets to standard values.

Parameter	Meaning
`\tabcolsep`	half the width of the spacing that is inserted between columns in the **tabular** and **tabular*** environments
`\arraycolsep`	the corresponding half intercolumn spacing for the **array** environment
`\arrayrulewidth`	the thickness of the vertical and horizontal lines within a table
`\doublerulesep`	the separation between two horizontal lines when `\hline\hline` is used

Changes in these parameters can be made with the `\setlength` command as usual.

Parameter	Meaning
`\arraystretch`	the factor by which the spacing between rows in an array is multiplied

A new value is set by redefining the parameter with the command,

```
\renewcommand{\arraystretch}{factor}
```

Somtimes, if the table is too simple, one can use `\hfill` like

```
\noindent\hfill
\begin{tabular}{...}
...
\end{tabular}
\hfill\mbox{}
```

or **center** environment to center the table on the page, and to prevent it from being left-aligned by the default.

[1] the top line of the table is aligned with the baseline of the current external line of text

[2] 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.

[3] This argument applies only to the tabular* environment. In this case, the *cols* argument must contain the @-expression `@{\extracolsep{\fill}}`.

[4] the text in this column is set into lines of width *wid*.

[5] the column format contained in cols is reproduced num times, so that *{5}{|c}| is the same as |c|c|c|c|c|.

[6] draws one or two vertical lines next to each other.

[7] inserts *text* in every line of the table between the two columns where it appears.

[8] draws one or two horizontal lines with 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.

[9] draws a horizontal line from the left side of column m to the right side of column n

[10] combines the following *num* columns into a single column. The argument *col* contains exactly one of the positioning symbols l, r, or c, with possible @-expressions and vertical lines |.

[11] draws a vertical line with the height of the row at the location where it appears.

6.2.3 Table examples

ex.1

1st Regional Soccer League — Final Results 2002/03							
	Club	*W*	*T*	*L*	*Goals*	*Points*	*Remarks*
1	Amesville Rockets	19	13	1	66:31	51:15	League Champs
2	Borden Comets	18	9	6	65:37	45:21	Trophy Winners
3	Clarkson Chargers	17	7	9	70:44	41:25	Candidates
4	Daysdon Bombers	14	10	9	66:50	38:28	for
5	Edgartown Devils	16	6	11	63:53	38:28	National
6	Freeburg Fighters	15	7	11	64:47	37:29	League
7	Gadsby Tigers	15	7	11	52:37	37:29	Medium Teams
8	Harrisville Hotshots	12	11	10	62:58	35:31	
9	Idleton Shovers	13	9	11	49:51	35:31	
10	Jamestown Hornets	11	11	11	48:47	33:33	
11	Kingston Cowboys	13	6	14	54:45	32:34	
12	Lonsdale Stompers	12	8	13	50:57	32:34	
13	Marsdon Heroes	9	13	11	50:42	31:35	
14	Norburg Flames	10	8	15	50:68	28:38	
15	Ollison Champions	8	9	16	42:49	25:41	
16	Petersville Lancers	6	8	19	31:77	20:46	Disbanding
17	Quincy Giants	7	5	21	40:89	19:47	Demoted
18	Ralston Regulars	3	11	19	37:74	17:49	

```
\begin{tabular}{|r|l||rrr|r@{:}l|r@{:}l||c|}
\hline
\multicolumn{10}{|c|}{\bfseries 1st Regional Soccer League --- Final Results
    2002/03} \\ \hline
& \itshape Club & \itshape W & \itshape T & \itshape L & \multicolumn{2}{c|}{\
    itshape Goals}
    & \multicolumn{2}{c||}{\itshape Points} & \itshape Remarks \\ \hline\hline
1 & Amesville Rockets & 19 & 13 & 1 & 66 & 31 & 51 & 15 & League Champs \\ \hline
2 & Borden Comets & 18 & 9 & 6 & 65 & 37 & 45 & 21 & Trophy Winners \\ \hline
3 & Clarkson Chargers & 17 & 7 & 9 & 70 & 44 & 41 & 25 & Candidates \\ \cline{1-9}
4 & Daysdon Bombers & 14 & 10 & 9 & 66 & 50 & 38 & 28 & for \\ \cline{1-9}
5 & Edgartown Devils & 16 & 6 & 11 & 63 & 53 & 38 & 28 & National \\ \cline{1-9}
6 & Freeburg Fighters & 15 & 7 & 11 & 64 & 47 & 37 & 29 & League \\ \hline
7 & Gadsby Tigers & 15 & 7 & 11 & 52 & 37 & 37 & 29 & \multirow{9}{*}{Medium Teams}
     \\ \cline{1-9}
8 & Harrisville Hotshots & 12 & 11 & 10 & 62 & 58 & 35 & 31 & \\ \cline{1-9}
9 & Idleton Shovers & 13 & 9 & 11 & 49 & 51 & 35 & 31 & \\ \cline{1-9}
10 & Jamestown Hornets & 11 & 11 & 11 & 48 & 47 & 33 & 33 & \\ \cline{1-9}
11 & Kingston Cowboys & 13 & 6 & 14 & 54 & 45 & 32 & 34 & \\ \cline{1-9}
12 & Lonsdale Stompers & 12 & 8 & 13 & 50 & 57 & 32 & 34 & \\ \cline{1-9}
13 & Marsdon Heroes & 9 & 13 & 11 & 50 & 42 & 31 & 35 & \\ \cline{1-9}
```

```
14 & Norburg Flames & 10 & 8 & 15 & 50 & 68 & 28 & 38 & \\ \cline{1-9}
15 & Ollison Champions & 8 & 9 & 16 & 42 & 49 & 25 & 41 & \\ \hline
16 & Petersville Lancers & 6 & 8 & 19 & 31 & 77 & 20 & 46 & Disbanding \\ \hline
17 & Quincy Giants & 7 & 5 & 21 & 40 & 89 & 19 & 47 & \multirow{2}{*}{Demoted} \\ \cline{1-9}
18 & Ralston Regulars & 3 & 11 & 19 & 37 & 74 & 17 & 49 & \\ \hline
\end{tabular}
```

In the textbook, The remark ‘Demoted’ is vertically placed in the middle of the last two rows. This is accomplished by typing

```
18  & Ralston Regulars & 3 & 11 & 19 & 37 &74 & 17 & 49
    & \raisebox{1.5ex}[0pt]{Demoted}\\ \hline
```

But the **multirow** package provides a more convenient way to do this, like:

```
17  & Quincy Giants & 7 & 5 & 21 & 40 & 89 & 19 & 47
    & \multirow{2}{*}{Demoted} \\ \cline{1-9}
18  & Ralston Regulars & 3 & 11 & 19 & 37 & 74 & 17 & 49
    & \\ \hline
```

ex.2

Budget Plan 2003–2004			
Project	Nr. □□□	Name □□□□□□□□□□□□	
Year	2003	2004	2005
	€ | $	€ | $	€ | $
Investment Costs			
Operating Costs			
Industrial Contracts			
Signature		Authorization	

```
\newsavebox{\boxk}\newsavebox{\boxkkk}
\sbox{\boxk}{\framebox[4mm]{\rule{0mm}{3mm}}}
\sbox{\boxkkk}{\usebox{\boxk}\usebox{\boxk}\usebox{\boxk}}

\begin{tabular}{|l|*{3}{>{\centering\arraybackslash}p{2.5cm}|}}
\hline
\multicolumn{4}{|c|}{\rule[-0.3cm]{0mm}{0.8cm}\bfseries Budget Plan 2003--2004}\\
\hline\hline
```

```
\rule[-0.4cm]{0mm}{1cm}Project
& \multicolumn{3}{l|}{Nr. \usebox{\boxkkk}\hspace{0.5cm}
    \vline\hspace{0.5cm}Name \usebox{\boxkkk}\usebox{\boxkkk}\usebox{\boxkkk}\
        usebox{\boxkkk}}\\ \hline
\multicolumn{1}{|r|}{Year} & 2003 & 2004 & 2005 \\
\cline{2-4}
& \euro\ \vline\ \$ & \euro\ \vline\ \$ & \euro\ \vline\ \$ \\ \hline
Investment & & & \\
Costs & & & \\ \hline
Operating & & & \\
Costs & & & \\ \hline
Industrial & & & \\
Contracts & & & \\ \hline
\multicolumn{4}{|l|}{\rule[-10mm]{0mm}{13mm}Signature \hspace{5cm}\vline ~
    Authorization} \\ \hline
\end{tabular}
```

Where I used the package **eurosym** to get the euro symbol, and the package **array** to use the `>{\centering\arraybackslash}` command to center the text in the p-columns (see next section). `\arraybackslash` is needed to restore the meaning of `\\` in the last column, which is changed by `>{\centering}`.

6.2.4 Extension packages for tables

array The **array** package adds several column formatting arguments, shown below:

Argument	**Meaning**
`m{wth}`	produces a column of width *wth* that is aligned vertically in the middle. (The standard pwth aligns the text with the top line.)
`b{wth}`	like `p` and `m` but aligns the text on the bottom line.
`>{decl}`	inserts *decl* before the next column.[12]
`<{decl}`	inserts *decl* after the last column.[13]
`!{decl}`	inserts *decl* between two columns without removing the normal intercolumn spacing, as for @decl.

With `\newcolumntype{name}[args]{definition}` one can define new column specifiers for multiple applications. For example, the command

```
\newcolumntype{P}[1]{>{\centering\arraybackslash}p{#1}}
```

defines a new column type **P** that takes an argument, which is the width of the column, and produces a p-column with centered text.

With `\firsthline` and `\lasthline`, horizontal lines can be issued before the first and after the last rows, respectively.

[12] ex., `>{\bfseries}` sets the entire column in boldface without having to type `\bfseries` in each row.

[13] to have a centered column in math mode, give `>{$}c<{$}`.

dcolumn To align numbers in columns on the decimal point, the **dcolumn** package provides a new column type **D** and loads the **array** package, its syntax is,

```
D{in_char}{out_char}{number}
```

where *in_char* is the character used in the input to separate the integer and fractional parts of the number, *out_char* is the character used in the output, and *number* is the maximum number of decimal places. For ex.,

projects	number A	number B
text 1	12.5	100.001
text 2	5.25	50.2
text 3	23.4	0.5

```
\begin{tabular}{l|d{2} d{3}}
projects & \multicolumn{1}{c}{number A}
         & \multicolumn{1}{c}{number B} \\ \hline
text 1 & 12.5  & 100.001 \\
text 2 & 5.25  & 50.2    \\
text 3 & 23.4  & 0.5     \\
\end{tabular}
```

If *number* is negative, the column is centered on the decimal point, otherwise it is right justified.

tabularx For example,

```
\begin{tabularx}{10cm}{c X l X} ... \end{tabularx}
```

defines a table with a total width of 10cm (like `tabular*`), The second and fourth columns use the `X` specifier, which creates expandable columns that automatically adjust *wth* to fill the remaining space, similar to a dynamic `p{wth}`.

delarray The **delarray** package allows you to add self-scaling delimiters to an **array** environment by placing them around the column format. For example:

```
\begin{array}({cc}) ... \end{array}
```

Here, the parentheses `(` and `)` function as `\left(` and `\right)`, automatically stretching to fit the height of the column data.

long-table The **longtable** package produces tables extending over several pages. It provides a **longtable** environment which takes the same column formatting argument as **tabular** and **array**, but has additional row entries at the start to determine:

1. those rows that only appear at the start of the table, terminated by `\endfirsthead`; this often includes the main `\caption`;
2. those at the top of every continuation page, terminated by `\endhead`; these normally include an additional `\caption` and the column headers;

3. those at the bottom of each page, terminated by `\endfoot`; and
4. those rows only at the end of the table, terminated by `\endlastfoot`.
5. normal table rows, which follow `\endlastfoot` and precede the end of the environment.

For ex.,

表 1: Demonstration of a long table

Left	Center	Right
Twenty-two	fifty	A hundred and eighty
22	50	180

end of table

```
\begin{longtable}{|l|c|r|}
\caption[Short title]{Demonstration of a long table}\\ % main caption
\hline
Left & Center & Right \\ % column headers, appear at the top of first page
\hline \endfirsthead
\caption[]{\emph{continued}}\\ % caption for continuation pages
\hline
Lef & Cent & Rig \\ % column headers for continuation pages
\hline \endhead
\hline
\multicolumn{3}{r}{\emph{continued on next page}} % footer for continuation pages
\endfoot
\hline
\multicolumn{3}{r}{\emph{end of table}} % footer for last page
\endlastfoot
Twenty-two & fifty & A hundred and eighty \\ % normal table rows
22 & 50 & 180 \\
\end{longtable}
```

6.2.5 Floating tables

See Section 8. Floating tables and figures.

7 Graphics Inclusion and Color

8 Floating tables and figures

9 Mathematical Formulas

9.1 Mathematical environments

For *inline text formulas*, use:

```
$ formula_text $
\( formula_text \)
\begin{math} formula_text \end{math}
```

All three are essentially the same, and there is no reason not to use the $ sign.

For *displayed formulas*, use:

```
\[ formula_text \]
$$ formula_text $$
\begin{displaymath} formula_text \end{displaymath}
\begin{euqation*} formula_text \end{equation*}
\begin{equation} formula_text \end{equation}
```

The differences between these is that the **equation** environment will automatically number the equations, while the others will not.

For *multi-line displayed formulas*, use:

```
\begin{eqnarray} formula_text \end{eqnarray}
\begin{eqnarray*} formula_text \end{eqnarray*}
```

However, the **eqnarray** environment is not recommended, as it has some spacing issues. Instead, consider using the **align** environment, which will be illustrated later.

9.2 Main elements

9.2.1 Constants and variables

Numbers that appear within formulas are called *constants*, whereas *simple variables* are represented by single letters. Usually, variables are set in *italics*, while constants are set in an upright font.

Blanks are totally ignored and are included in the input text simply to improve the appearance for the writer. To insert a space in a formula, use `\␣`, `~`, or `\quad`, `\qquad`, etc.

The curly braces { } serve the purpose of logically combining parts of the formula and therefore cannot act as printable characters. To print a curly brace, use `\{` or `\}`.

9.2.2 Exponents and indices

`a^b`	a^b
`a_b`	a_b
`a^{b+c}`	a^{b+c}
`a_{b+c}`	a_{b+c}

Note: The raising and lowering commands ^and _ are only permitted in math mode.

9.2.3 Fractions

`\frac{numerator}{denominator}`	$\frac{numerator}{denominator}$

9.2.4 Roots

`\sqrt[n]{arg}`	$\sqrt[n]{arg}$

9.2.5 Sums and integrals

In a text formula:

`\sum_{i=1}^{n}`	$\textstyle\sum_{i=1}^{n}$
`\sum\nolimits_{i=1}^{n}`	$\textstyle\sum\limits_{i=1}^{n}$
`\int_{a}^{b}`	$\textstyle\int_{a}^{b}$
`\int\limits_{a}^{b}`	$\textstyle\int\limits_{a}^{b}$

In a displayed formula:

`\sum_{i=1}^{n}`	$\displaystyle\sum_{i=1}^{n}$
`\sum\nolimits_{i=1}^{n}`	$\displaystyle\sum\nolimits_{i=1}^{n}$
`\int_{a}^{b}`	$\displaystyle\int_{a}^{b}$
`\int\limits_{a}^{b}`	$\displaystyle\int\limits_{a}^{b}$

Here the `\nolimits` and `\limits` commands control the placement of the limits of summation and integration.

Two points must be made regarding integrals. First, there should be a little extra spacing between the differential operators dx and dy and the preceding part of the formula. This is achieved by using the small space command `\,`. Second, the differential operators should be set in an upright font, which is achieved by using the `\mathrm` command as follows:

```
\[
\int_{a}^{b} f_i(x,y)g_i(x,y)\,\mathrm{d}x\,\mathrm{d}y
\]
```

$$\int_{a}^{b} f_i(x,y)g_i(x,y)\,\mathrm{d}x\,\mathrm{d}y$$

9.2.6 Continuation dots

`\ldots`	$\ldots$	*low dots*	`\cdots`	$\cdots$	*center dots*
`\vdots`	$\vdots$	*vertical dots*	`\ddots`	$\ddots$	*diagonal dots*

The command `\ldots` is also available in normal text mode, whereas the other three are only allowed in math mode. In text mode, the command `\dots` may be used in place of `\ldots` with the same effect.

9.3 Mathematical symbols

9.3.1 Greek letters

LaTeX normally sets the uppercase Greek letters in Roman (upright) type within a mathematical formula. If they need to be in *italics*, this can be done with the math alphabet command `\mathnormal`: `$\mathnormal{\Gamma\Pi\Phi}$` appears as $\mathnormal{\Gamma\Pi\Phi}$.

lowercase 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`				

Uppercase letters

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

9.3.2 Calligraphic letters

`\mathcal{A, B, C, ...}`	$\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$

9.3.3 Binary operators

Two mathematical quantities combined with one another to make a new quantity are said to be joined by a *binary operation.*

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

The underlined commands are defined in one of the **amssymb**, **amsfonts**, or **latexsym** packages.

9.3.4 Relations and negated relations

Relations

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

negated relations

$\not<$	`\not<`	$\not>$	`\not>`	$\not=$	`\not= neq`
$\not\leq$	`\not\leq`	$\not\geq$	`\not\geq`	$\not\equiv$	`\not\equiv`
$\not\prec$	`\not\prec`	$\not\succ$	`\not\succ`	$\not\sim$	`\not\sim`
$\not\preceq$	`\not\preceq`	$\not\succeq$	`\not\succeq`	$\not\simeq$	`\not\simeq`
$\not\subset$	`\not\subset`	$\not\supset$	`\not\supset`	$\not\approx$	`\not\approx`
$\not\subseteq$	`\not\subseteq`	$\not\supseteq$	`\not\supseteq`	$\not\cong$	`\not\cong`
$\not\sqsubseteq$	`\not\sqsubseteq`	$\not\sqsupseteq$	`\not\sqsupseteq`	$\not\asymp$	`\not\asymp`
$\not\in$	`\not\in`	$\notin$	`\notin`		

9.3.5 Arrows

$\leftarrow$	`\leftarrow \gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\rightarrow$	`\rightarrow \to`	$\longrightarrow$	`\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`		

9.3.6 Various other symbols

<table>
<tr><td>$\aleph$</td><td><code>\aleph</code></td><td>$\prime$</td><td><code>\prime</code></td><td>$\forall$</td><td><code>\forall</code></td><td>$\Box$</td><td><code>\Box</code></td></tr>
<tr><td>$\hbar$</td><td><code>\hbar</code></td><td>$\emptyset$</td><td><code>\emptyset</code></td><td>$\exists$</td><td><code>\exists</code></td><td>$\Diamond$</td><td><code>\Diamond</code></td></tr>
<tr><td>$\imath$</td><td><code>\imath</code></td><td>∇</td><td><code>\nabla</code></td><td>$\neg$</td><td><code>\neg</code></td><td>$\triangle$</td><td><code>\triangle</code></td></tr>
<tr><td>$\jmath$</td><td><code>\jmath</code></td><td>$\surd$</td><td><code>\surd</code></td><td>$\flat$</td><td><code>\flat</code></td><td>$\clubsuit$</td><td><code>\clubsuit</code></td></tr>
<tr><td>ℓ</td><td><code>\ell</code></td><td>∂</td><td><code>\partial</code></td><td>$\natural$</td><td><code>\natural</code></td><td>$\diamondsuit$</td><td><code>\diamondsuit</code></td></tr>
<tr><td>$\wp$</td><td><code>\wp</code></td><td>$\top$</td><td><code>\top</code></td><td>$\sharp$</td><td><code>\sharp</code></td><td>$\heartsuit$</td><td><code>\heartsuit</code></td></tr>
<tr><td>$\Re$</td><td><code>\Re</code></td><td>$\bot$</td><td><code>\bot</code></td><td>$\|$</td><td><code>\|</code></td><td>$\spadesuit$</td><td><code>\spadesuit</code></td></tr>
<tr><td>$\Im$</td><td><code>\Im</code></td><td>$\vdash$</td><td><code>\vdash</code></td><td>$\angle$</td><td><code>\angle</code></td><td>$\Join$</td><td><code>\Join</code></td></tr>
<tr><td>$\mho$</td><td><code>\mho</code></td><td>$\dashv$</td><td><code>\dashv</code></td><td>$\backslash$</td><td><code>\backslash</code></td><td>∞</td><td><code>\infty</code></td></tr>
</table>

9.3.7 Variable-sized symbols

The following symbols are printed in different sizes depending on whether they appear in text or displayed formulas:

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

One can also use `\displaystyle` before these symbols in text formulas to make them appear in their larger size, but this is not recommended as it disrupts the line spacing.

9.3.8 Function names

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

and the function mod in one of two forms:

`\$ a \bmod b \$`	⇒	$a \bmod b$
`\$ y \pmod{a+b} \$`	⇒	$y \pmod{a+b}$.

9.3.9 Accents

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

The letters i and j should be printed without their dots when they are given an accent. To accomplish this, type the symbols `\imath` and `\jmath` instead of the letters, as in

`$\vec{\imath} + \tilde{\jmath}$` $\vec{\imath} + \tilde{\jmath}$

There are wider versions of `\hat` and `\tilde` available:

`$\widehat{1-x}=\widehat{-y}$`	$\widehat{1-x}=\widehat{-y}$
`$\widetilde{xyz}$`	$\widetilde{xyz}$

9.4 Additional elements

9.4.1 label and ref

label and **ref** are used to refer to equations, figures, tables, etc., like:

```
\begin{equation}
E=mc^2
\label{eq:emc2}
\end{equation}
...
As shown in equation~\ref{eq:emc2}, ...
```

$$E = mc^2 \tag{1}$$

...

As shown in equation 1, ...

9.4.2 Automatic sizing of brackets

`\left` *lbrack* *sub_form* `\right` *rbrack*

The *lbrack* and *rbrack* can be any of the following:

(	`(`	)	`)`	⌊	`\lfloor`	⌋	`\rfloor`
[	`[`	]	`]`	⌈	`\lceil`	⌉	`\rceil`
{	`\{`	}	`\}`	⟨	`\langle`	⟩	`\rangle`
|	`|`	‖	`\\|`	↑	`\uparrow`	⇑	`\Uparrow`
/	`/`	\\	`\backslash`	↓	`\downarrow`	⇓	`\Downarrow`
				↕	`\updownarrow`	⇕	`\Updownarrow`

To print an invisible bracket, use `\left.` or `\right.`.

9.4.3 Ordinary text within formulas

It is often necessary to include some normal text within a formula, for example, single words such as *and*, *or*, *if*, and so on. In such cases, the `\text` command from the **amsmath** package is recommended[1], as it will ensure that the text is set in the same font as the surrounding text.

On the other hand, if letters from text fonts are required as mathematical symbols, they should be entered with the *math alphabet commands*:

```
\mathrm     \mathtt     \mathbf
\mathsf     \mathit     \mathnormal     \mathcal
```

9.4.4 Arrays

The syntax for a matrix has been given in 6.2 Tabular, it's:

```
\begin{array}[pos]{cols}      rows     \end{array}
```

Here only show an example of an array:

$$\sum_{p_1<p_2<\cdots<p_{n-k}}^{(1,2,\ldots,n)} \Delta \begin{array}{c} \\ p_1p_2\cdots p_{n-k} \\ p_1p_2\cdots p_{n-k} \end{array} \sum_{q_1<q_2<\cdots q_k} \left| \begin{array}{cccc} a_{q_1q_1} & a_{q_1q_2} & \cdots & a_{q_1q_k} \\ a_{q_2q_1} & a_{q_2q_2} & \cdots & a_{q_2q_k} \\ \multicolumn{4}{c}{\dotfill} \\ a_{q_kq_1} & a_{q_kq_2} & \cdots & a_{q_kq_k} \end{array} \right|$$

[1] `\mbox` is an older command that can also be used, but it does not allow for line breaks and may not work well with certain fonts.

```
\[
\sum_{p_1<p_2<\cdots<p_{n-k}}^{(1,2,\ldots,n)}
\Delta_{\begin{array}{l}
p_1p_2\cdots p_{n-k} \\ p_1p_2\cdots p_{n-k}
\end{array}}
\sum_{q_1<q_2<\cdots q_k} \left| \begin{array}{llcl}
a_{q_1q_1} & a_{q_1q_2} & \cdots & a_{q_1q_k} \\
a_{q_2q_1} & a_{q_2q_2} & \cdots & a_{q_2q_k} \\
\multicolumn{4}{c}{\dotfill}\\
a_{q_kq_1} & a_{q_kq_2} & \cdots & a_{q_kq_k}
\end{array} \right|
\]
```

9.4.5 Lines above and below formulas

Sometimes it is necessary to add lines above or below a formula. This can be done using the `\overline` and `\underline` commands:

$$\overline{a+b} \quad \underline{a+b}$$

The command `\underline` is also available in text mode, as in `\underline{a}`, which produces $\underline{\text{a}}$.

Alternatively, the `\overbrace` and `\underbrace` commands can be used to add braces above or below a formula:

$$\overbrace{a+b}^{\text{sum}} \quad \underbrace{a+b}_{\text{sum}}$$

9.4.6 Stacked symbols

The `\stackrel` command can be used to stack one symbol above another, like this:

$$a \stackrel{f}{\longrightarrow} b$$

The syntax is `$\stackrel{top_sym}{bottom_sym}$`, which places the *top_sym* symbol centered on the top of *bottom_sym*.

9.4.7 \atop and \above

The syntax of the TeX commands `\atop` and `\choose` is as follows:

```
{top \atop bottom}
{top \choose{dimen} bottom}
```

which is similar to,

```
\begin{array}{c}  upper_line \\ lower_line  \end{array}          (atop)
\left(\begin{array}{c} upper \\ lower \end{array}\right)        (choose)
```

The `\atop` command is useful for double summations,

```
\[
\sum_{k_0,k_1\ldots\ge0 \atop k_0+k_1+\cdots=0}
\]
```

$$\sum_{k_0,k_1\ldots\ge0 \atop k_0+k_1+\cdots=0}$$

`\atop` is preferred because the indices created by **array** environment appear too large, whereas the `\atop` have varying sizes depending on the context, and they are more compact.

The `\choose` command is useful for binomial coefficients:

```
\[
{n+1 \choose k} = {n \choose k} + {n \choose k-1}
\]
```

$${n+1 \choose k} = {n \choose k} + {n \choose k-1}$$