Dilip Datta

LaTeX in 24 Hours

A Practical Guide for Scientific Writing

Springer

LaTeX in 24 Hours

A Practical Guide for Scientific Writing

Dilip Datta

LaTeX in 24 Hours

A Practical Guide for Scientific Writing

Springer

Dilip Datta
Department of Mechanical Engineering
Tezpur University
Tezpur, Assam
India

ISBN 978-3-319-47830-2 ISBN 978-3-319-47831-9 (eBook)
DOI 10.1007/978-3-319-47831-9

Library of Congress Control Number: 2016956633

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To My Parents
who gifted me the life

Preface

The necessity for writing this book was felt long back, during my Ph.D. work, when I saw students and researchers struggling with LaTeX for preparing their articles and theses. A very limited number of books on LaTeX are available in markets. Of course, a lot of resources on this subject can be obtained freely from the internet. However, most of the books emphasize on detailed documentation of LaTeX, while the internet-based resources are topic-specific. But people are either unable or not interested to spare time, during their busy schedules of research works, to understand and learn the detailed genotype of LaTeX covered in books, or to collect materials from different websites. Instead, they prefer to get direct and concise applications of various LaTeX syntax in a single window, which they can modify easily, so as to get their works done in the least time and with the least effort. This is the motivation for writing the book. The book has been prepared by following a huge number of existing books and internet-based resources, as well as my personal experience with LaTeX (but the Bibliographic list has been shortened referring only to some famous resources). An attempt is made here to present materials in such a way that, at least, a similar book can be produced using only this book. *Using only the raw version of this book, many of my students have already learned* LaTeX *successfully up to the level of preparing articles and dissertations*. Hence, I am confident that the book would be able to cater to the needs of other students and professionals also, who want to learn and use LaTeX in a short time. Suggestions for any correction, modification, addition, or deletion will be highly appreciated (the same may be mailed to ddatta@tezu.ernet.in, datta_dilip@rediffmail.com or ddatta@iitkalumni.org).

The book, **LaTeX in 24 Hours: A Practical Guide for Scientific Writing**, explains the basic LaTeX2ε required for writing scientific documents. Applications of most of the discussed LaTeX syntax are presented in such a way that a reader would be able to use them directly without any confusion, however maybe with some minor modifications as per requirement. In many cases, multiple procedures are presented for producing a single item. The main part of the book is stretched over 276 pages dividing into 24 chapters, named as Hours. Hour 1 introduces LaTeX, including how a LaTeX document is prepared and compiled. Various LaTeX syntax required for fonts selection, texts and page formatting, items listing, table preparation, figure insertion and drawing, equation writing, user-defined macros, bibliography preparation, list of contents and index generation, and some other miscellaneous issues are discussed in Hours 2–18. Hours 19 and 20 explain the preparation of complete documents, such as letter, article, book, and report. Since a work often needs to be presented to an audience, slide preparation is also explained in Hours 21 and 22. Being an unavoidable fact, error and warning messages generated in different cases are discussed in Hour 23. Finally, some exercises are included for learners in Hour 24. Further, LaTeX commands for producing different symbols are presented in Appendix A.

I am thankful to my Ph.D. supervisor, Prof. Kalyanmoy Deb, Indian Institute of Technology Kanpur, India (presently in Michigan State University, USA), from whom I could learn many things about LaTeX. In fact, I was inspired to work with LaTeX from him only. I am also thankful to my friend, Dr. Shamik Choudhury (working in GE Capital, Bengaluru, India), who helped me in preparing and compiling my very first LaTeX document. My special thanks are due to my better half, Madhumita, and beloved daughters, Devoshree and Tanushree, from whom I got continuous inspiration and support for writing the book. Finally, I am indebted to my parents, late Paresh Chandra Datta and late Saraju Datta, whose blessings have brought me to this height to be able to write a book.

Tezpur, India Dilip Datta
September 2016

About the Author

Dilip Datta obtained his Bachelor degree in Mechanical Engineering from Gauhati University, Master degree in Applied Mechanics from The Indian Institute of Technology (IIT) Delhi, and Ph.D. in Optimization from IIT Kanpur. He has been teaching Mechanical Engineering courses, as well as some interdisciplinary courses for more than twenty years. His research area is optimization, specially evolutionary algorithms for multi-objective combinatorial optimization problems. However, it is his passion to play with LaTeX. This book is written based on his personal experience with LaTeX. Going through the initial draft of this book only, many of his students could learn LaTeX up to the level of writing scientific articles and academic theses. Hence, he believes that this book will be a proper practical guide for beginners to learn LaTeX.

Contents

Page

List of Tables

List of Figures

Hour 1

Introduction

1.1 What Is LaTeX?

Donald E. Knuth developed TeX in the year 1977 as a typesetting system for preparing books, especially those containing a lot of mathematical expressions. Based on it, Leslie Lamport developed LaTeX (named as LaTeX 2.09) in 1985 for preparing documents by concentrating on the structure of a document rather than on its formatting details. LaTeX 2.09 was enhanced in 1994 as LaTeX 2_ε by a group of developers led by Frank Mittelbach.

LaTeX is a macro-package used as a language-based approach for typesetting documents. Various LaTeX instructions are interspersed with the input file of a document, say `myfile.tex`, for obtaining the desired output as `myfile.dvi` or directly as `myfile.pdf`. The `myfile.dvi` file can be used to generate `myfile.ps` or even `myfile.pdf` file. However, unlike programming languages for computational works, such as `C` or `C++`, LaTeX is very simple and easy to work with. One can become expert in LaTeX through a little practice. LaTeX can be used for preparing letters, applications, articles, reports, publications, theses, books, or anything of that kind.

1.2 Why LaTeX Over Other Word Processors?

The use of common word processors, in which the effect becomes directly visible, may be easier in preparing simple and small documents. But, as shown roughly in Fig. 1.1, the effort and time required in LaTeX for preparing complicated and big-size documents are quite less than those required in other word processors[1]. LaTeX is especially well suited

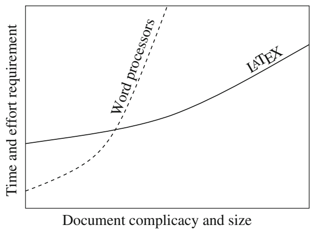


Fig. 1.1 LaTeX and other word processors

[1]Effort and time required in LaTeX for preparing complicated and big-size documents are quite less than those required in other word processors.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_1

for scientific writing, like technical reports, articles, academic dissertations, books, etc. Although learning LaTeX is time and effort taking, it can be realized that the preparation of only one academic dissertation would pay off all additional efforts required in learning LaTeX.

One of the major advantages of using LaTeX is that manual formatting of a document, as usually required in many word processors, can be automated in LaTeX. Therefore, the possibility of doing any mistake in numbering and referring items (sections, tables, figures or equations), in choosing size and type of fonts for different sections and subsections, or in preparing bibliographic references, can be avoided. Further, LaTeX has the provision for automatically generating various lists of contents, index, and glossary.

1.3 How to Prepare a LaTeX Input File?

As shown in Fig. 1.2, the main structure of a LaTeX input file can be divided into two parts – *preamble* and *body*.

```
\documentclass[]{}      } Preamble
.
.
\begin{document}        }
.                       }
.                       } Body
.                       }
.                       }
\end{document}          }
```

Fig. 1.2 Structure of a LaTeX input file

The preamble is the first part of an input file that contains the global processing parameters for the entire document to be produced, such as the type of the document, page formatting, header and footer setting, inclusion of LaTeX packages for supporting additional instructions, and definitions of new instructions. The simplest preamble is **\documentclass{**`dtype`**}**, where `dtype` in **{}** is a mandatory argument as the class (or type) of the document, such as **letter**, **article**, **report**, or **book**[2]. In the default setting, **\documentclass{}** prints a document on letter-size paper in 10 point fonts (1 point ≈ 0.0138 inch ≈ 0.3515 mm). Different user-defined formats for a document can be obtained through various options to **\documentclass{}**[3], in which case it takes the form of **\documentclass[**`fo1`**,**`fo2`**,...]{**`dtype`**}** with `fo1`, `fo2`, etc., in **[]** as the options (multiple options can be inserted in any order separating two options by a comma), e.g., **\documentclass[a4paper,11pt]{article}** for printing an article on A4 paper in 11 point fonts. Some standard options to **\documentclass[]{}** are listed in Table 1.1 on the next page.

As shown in Fig. 1.2, the main body of a LaTeX input file starts with **\begin{document}** and ends with **\end{document}**. The entire contents to be printed in the output are inserted within the body, mixed with various LaTeX instructions (see §1.5 on page 5 for details). Any text entered after **\end{document}** is simply skipped by a LaTeX compiler.

---

[2]The standard document classes are **letter**, **article**, **report** and **book**. Besides these four standard classes, some other classes are also available, such as **amsart**, **thesis**, **slide**, **slides** and **seminar**.

[3]Different user-defined formats for a document can be obtained through various options to **\documentclass[]{}**.

Table 1.1 Standard options to the **\documentclass[ ]{ }** command

Format	Options	Function
Font size	**10pt** (default), **11pt** and **12pt**	Self-explanatory
paper size	**letterpaper** (default), **a4paper**, **a5paper**, **b5paper**, **legalpaper** and **executivepaper**	Self-explanatory
Page orientation	**portrait** (default) and **landscape**	Self-explanatory
Columns of texts	**onecolumn** (default) and **twocolumn**	Self-explanatory
Type of printing	**oneside** (default for **article** and **report**) and **twoside** (default for **book**)	Self-explanatory
New chapter	**openright** (default for **book**) **openany**	Starts on the next odd page Starts on the next page
Title printing	**titlepage** and **notitlepage**	Self-explanatory
Equation	**leqno** **fleqn**	Number on left hand side Equations are left aligned
Drafting	**final** (default) and **draft**	Self-explanatory
Bibliography	**openbib**	A part on a separate line

A LaTeX input file is named with `tex` extension, say `myfile.tex`. It can be prepared in any operating system using any general-purpose text editor that supports `tex` extension, e.g., gedit or Kate in Linux-based systems. There are also many open-source (free access in internet) text editors developed specifically for preparing LaTeX input files, e.g.,

▷ *For both Windows and Linux*:

- BaKoMa TeX (http://bakoma-tex.com)
- Emacs (www.gnu.org/software/emacs/emacs.html)
- jEdit (http://jedit.org)
- Kile (http://kile.sourceforge.net)
- LyX (www.lyx.org)
- Open LaTeX Studio (http://sebbrudzinski.github.io/Open-LaTeX-Studio)
- TeXlipse (http://texlipse.sourceforge.net)
- TeXmacs (www.texmacs.org)
- Texmaker (www.xm1math.net/texmaker)
- TeXpen (https://sourceforge.net/projects/texpen)
- TeXstudio (www.texstudio.org)
- TeXworks (https://github.com/TeXworks/texworks), etc.

▷ *For Linux only*:

- LaTeXila (https://wiki.gnome.org/Apps/LaTeXila)
- Gummy (https://github.com/alexandervdm/gummi), etc.

▷ *For Windows only*:

- Inlage (www.inlage.com)
- LEd (www.latexeditor.org)
- TeXnicCenter (www.texniccenter.org)
- WinShell (www.winshell.de)
- WinEdt (www.winedt.com), etc.

Table 1.2 A simple LaTeX input file and its output

LaTeX input	Output
\documentclass{article} \begin{document} **LaTeX** is a macro package for typesetting documents. It is a language-based approach, where LaTeX *instructions* are interspersed with the text file of a document, say `myfile.tex`, for obtaining the desired output as `myfile.dvi`. The `myfile.dvi` file can then be used to generate `myfile.ps` or `myfile.pdf` file. \end{document}	LaTeX is a macro package for typesetting documents. It is a language-based approach, where LaTeX instructions are interspersed with the text file of a document, say myfile.tex, for obtaining the desired output as myfile.dvi. The myfile.dvi file can then be used to generate myfile.ps or myfile.pdf file. 1

A simple input file, say `myarticle.tex`, prepared under the document-class of **article** is shown in the left column of Table 1.2, along with its output in the right column. The actual contents to be printed in the output file were inserted within **\begin{document}** and **\end{document}**. Surprisingly, the output is not the one as expected. The differences are shown underlined in the output file. This has happened due to the fact that many things in LaTeX can be obtained through some special instructions only as stated in §1.5. Another thing to be noticed is that the output file is assigned a default page number at the bottom-center. However, before going to such issues, the compilation procedure of a LaTeX file is stated first in the following section.

1.4 How to Compile a LaTeX Input File?

A LaTeX input file can be compiled in many LaTeX editors mentioned in § 1.3 on page 2. Besides, operating-system based many open-source LaTeX compilers are also available, e.g., MiKTeX (http://miktex.org) for Windows operating system, or TeXLive (https://www.tug.org/texlive) for both Windows- and Linux-based operating systems. In a GUI-based compiler, like MiKTex or Kile, a LaTeX file can be compiled just by a mouse-click. In other command-line compilers, a LaTeX file is to be compiled through the latex command, followed by the name of the input file with or without its `tex` extension. For example, `myarticle.tex` of Table 1.2 can be compiled as follows:

```
$ latex myarticle
```

This command will produce three files, namely `myarticle.aux`, `myarticle.log`, and `myarticle.dvi` (refer § 20.4.1 on page 199 for detail). Out of these three files, `myarticle.dvi` is the final output which can be viewed in a document viewer, such as *xdvi* or *Evince*. The `myarticle.dvi` file can also be used for producing `myarticle.ps` or `myarticle.pdf` as the final output of `myarticle.tex`. The commands for these are as follows:

```
    $ dvips -o myarticle.ps  myarticle.dvi
Or, $ dvipdf myarticle.dvi
```

A pdf file can also be generated directly using the **pdflatex** command, instead of the **latex** command as stated above. For example, the following command will also generate myarticle.pdf:

$ pdflatex myarticle.tex

The compilation will need some more commands if a document involves bibliography, lists of contents, index, etc. Such commands will be addressed in relevant Hours.

1.5 LaTeX Syntax

LaTeX syntax consists of commands and environments, which are kinds of instructions interspersed with the texts of a document for performing some specific jobs. Such instructions are defined in different packages[4].

1.5.1 *Commands*

A LaTeX command is an independent instruction used either for producing something new or to change the form of an existing item, e.g., producing the symbol α or printing `italic` as *italic*. Different properties of the LaTeX commands are stated below:

- ▷ A command usually starts with a **** (backslash), followed by one or more characters without any gap in between. For example, **\LaTeX** and **\copyright** for producing LaTeX and © respectively.
- ▷ Non-alphabetic characters normally cannot appear in the name of a LaTeX command. Some LaTeX internal commands may start as **\@**, which (commands) are to be put in the preamble preceded and followed, respectively, by the **\makeatletter** and **\makeatother** commands.
- ▷ Many commands require some mandatory arguments, up to a maximum limit of nine, each in a separate pair of **{}** (preferably without any gap in between), e.g., **\textcolor{blue}{**this is blue colored**}** (detail is in §2.4 on page 13) for printing the second argument in blue color.
- ▷ Many commands have the provision for accepting some optional instructions also, written in **[]** separating two options (instructions) by a comma, e.g., **\documentclass[a4paper,11pt,twoside]{article}**. A command with the provision for optional arguments must have at least one mandatory argument.
- ▷ A command ended by an alphabet (i.e., a command not having any argument) ignores trailing blank spaces. Hence, if followed by a word or a number, such a command should be ended by **\␣** (the ␣ symbol is used to indicate a blank

[4] In this book, LaTeX commands, environments and packages, as well as other LaTeX syntax, are printed in **red colored** (for online version) **and boldfaced sans serif fonts** for their clear distinction from other texts of the book.

space obtained by pressing the Spacebar or Tab button of the keyboard). For example, \copyright␣2007 will produce ©2007, while \copyright\␣2007 will produce © 2007. However, if such a command is followed by any punctuation, it needs not to be ended by \␣ as a punctuation is not to be preceded by any blank space.

1.5.2 Environments

A LaTeX environment is a structure composed of two complementary commands, within which some particular job can be performed, e.g., writing an equation or inserting a figure. Different properties of the LaTeX environments are as follows:

▷ The pair of complementary commands creating an environment structure are \begin{ename} and \end{ename}, where ename is the name of the environment, e.g., \begin{document} and \end{document} as shown in Fig. 1.2 on page 2 creates the document environment (or the body) in a LaTeX input file.

▷ It is possible to use a command inside an environment, or to nest two or more environments, e.g., within the document environment, the \LaTeX command for printing LaTeX or the figure environment for inserting an external figure.

▷ Many environments require some mandatory arguments, which are placed after \begin{}, e.g., \begin{spacing}{1.3} for creating a line spacing of 1.3 pt through the spacing environment, or \begin{tabularx}{10cm}{XXX} for creating a table of three equal-width columns over 10 cm length through the tabularx environment.

▷ Like a command, many environments also have the provision for accepting some optional instructions written in a pair of [ ], e.g., \begin{table}[t] preferring through the option t in the table environment to place a table at the top of a page.

1.5.3 Packages

The class (or type) of a document, incorporated through the mandatory argument of the \documentclass{} command, includes some basic features of the document, like page layout and sectioning. Provision is also there to invoke additional commands and environments in a document for adding extra features that are not parts of the standard document class. Such commands and environments are defined in separate files, known as packages.

▷ A package is loaded (included) in the preamble, in between the \documentclass{} and \begin{document} commands[5], through the \usepackage{pname} command, where pname is the name of the package, e.g., \usepackage{color} for producing colored texts or \usepackage{amssymb,amsmath} for producing AMS type mathematical symbols and expressions.

---

[5] If anything (a command or a package) is asked to be put in the preamble of a LaTeX input file, it is to be inserted in between the \documentclass{} and \begin{document} commands.

▷ Like commands and environments, many packages also have the provision for accepting some optional instructions in **[ ]**, e.g., **\usepackage[tight]{subfigure}** preferring through the option **tight** to reduce extra space between figures.

▷ Unlike an option to **\documentclass[ ]{ }**, which is global to the entire document – including other packages too, an option to **\usepackage[ ]{ }** is local only to the features defined in the package(s) loaded through the **\usepackage[ ]{ }** command.

1.6 Keyboard Characters in LaTeX

Not all, but only those characters of an *English keyboard*[6], shown in Table 1.3, can be

Table 1.3 Keyboard characters that can be produced directly

Type of character	Characters
Uppercase letters	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Lowercase letters	a b c d e f g h i j k l m n o p q r s t u v w x y z
Digits	0 1 2 3 4 5 6 7 8 9
Parentheses	( )
Brackets	[ ]
Quotations	` ' "
Punctuation	, ; : ! . ?
Math operators	+ - * / =
Other symbols	@

printed directly in a LaTeX document (the left-hand quotation mark (`), shown in Table 1.3, generally appears on the same button with the ∼ symbol). All other characters of an English keyboard, which are not included in Table 1.3, need to be produced in a LaTeX document through some commands (most of these characters are reserved in LaTeX for special purposes). Table 1.4 on the following page lists those special characters, commands for producing them in a LaTeX document, and also the purposes, if any, for which these are reserved in LaTeX. For the commands starting and ending with the **$** symbol (i.e., in **$ $**), the **amssymb** package may be required (when used in an equation, as addressed in Hour 11 on page 101, these commands need not to be enclosed in **$ $**).

The special keyboard characters, listed in Table 1.4, can be produced in text-mode using the **\verb" "** or **\verb! !** command also, e.g., **\verb**"$" for printing $, or **\verb!**∼**!** for printing ˜ (the **\verb" "** and **\verb! !** commands are used for printing *as-it-is* what is written within **" "** or **! !** in the input file).

The commands for producing language-specific keyboard characters, as well as many other characters and mathematical symbols, are given in Appendix A on page 247.

---

[6]Many computers are manufactured for particular countries where a keyboard contains some language-specific characters, in addition to those used in the English language. However, for general purpose, a keyboard containing the characters, used in the English language only, will be discussed in this book.

Table 1.4 Keyboard characters to be produced through commands

Character	Command	Function in LaTeX
$	**\$**	A pair of $ creates a math-mode* within text-mode.
%	**\%**	Texts of a line preceded by % are commented.
{ }	**\{ \}**	Mandatory arguments of a command are written within **{}**.
_	**_**	Generates a subscript in math-modes.
ˆ	**\ˆ**	Generates a superscript in math-modes.
&	**\&**	Separates the entries of two columns in a Table.
#	**\#**	Miscellaneous symbol.
\	**$\backslash$**	Most of the LaTeX commands start with ****.
∼	**$\sim$**	Binds two words to be printed in the same line.
|	**$\|$**	Generates a vertical (column) line in a Table.
<	**$<$**	—
>	**$>$**	—

*Text processing modes are discussed in the first paragraph of Hour 2 on the next page.

1.7 How to Read this Book?

The version LaTeX 2_ε of LaTeX is discussed in this book. However, without referring the version, only the general term LaTeX is used throughout the book.

It is not a reference manual, but a practical guide prepared based only on the author's own experiences. The book is intended for beginners, and hence it discusses mainly the basic elements available in the standard LaTeX 2_ε distribution. Many relevant advanced topics are also discussed marking them as starred (*), which can be skipped in the initial stage of learning. Many statements are repeated in the book so that a reader is not necessarily required to remember everything. All LaTeX syntax, including commands and environments, are printed in **red colored** (for online version) **and boldfaced sans serif fonts** to make them easily distinguishable from normal texts.

Since the book is meant for beginners, it is suggested to read the entire book for better understanding of LaTeX. However, as mentioned in the Preface that many students and professionals are interested to get their works done in less time and least efforts, one can move directly to Hour 19 on page 181 or Hour 20 on page 191 to start writing a document immediately. Then other Hours can be followed for required additional information, where different topics – writing equations, drawing tables, inserting figures, etc., are discussed in detail. Also, many important points are highlighted in foot notes on various pages. For quickly locating the availability of different topics, one may browse through the `Contents`, `List of Tables` and `List of Figures` at the beginning of the book, or `Index` at the end of the book for searching different terminologies, where attempts are made to include information almost about all the materials that are discussed in this book.

Hour 2

Fonts Selection

There are three modes for processing texts in LaTeX – *paragraph*-mode, *LR*-mode and *math*-mode[1]. The paragraph-mode is for producing normal texts with automatic word-splitting, and line and page breaking to fit the texts within the area specified by the width and height of a page. In contrast, the LR-mode processes texts from left-to-right without any word-splitting and line breaking, such as **\mbox{}** or **\fbox{}** command whose arguments may span even beyond the specified width of a page. On the other hand, the math-mode is for writing mathematical expressions, like equations. In this book, the paragraph-mode and LR-mode will occasionally be addressed by a single name, known as the *text*-mode.

2.1 Text-Mode Fonts

The default font type of a LaTeX document is medium series serif family in upright shape and 10 pt size. The sizes of fonts in different parts of a document, say in headings and in paragraphs, are calculated proportionately, which can be visualized in this book. The default font setting can be altered globally through various options to the **\documentclass[ ]{}** command, e.g., **\documentclass[**`12`**pt]{article}** for producing an article in 12 pt fonts. The type of fonts in a particular segment can also be set manually as discussed below.

Types of fonts in LaTeX are classified into four categories – *family*, *series*, *shape* and *size*. The detail of each category is given in Table 2.1 on the following page.

1. *Font family*: There are three standard font families, namely *serif* (default), *sans serif* and *typewriter* fonts, which are accessed by the **\textrm{}** (or **{\rm }**), **\textsf{}** (or **{\sf }**) and **\texttt{}** (or **{\tt }**) commands respectively. The same can also be accessed by the **\rmfamily**, **\sffamily** and **\ttfamily** declarations, respectively.

[1] LaTeX processes texts in three modes – paragraph-mode, LR-mode and math-mode.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_2

Table 2.1 Different types of text-mode fonts used in LaTeX

SN	Type	Variety	Command	Declaration
1	Family	Serif family (default)	\textrm{atext} or {\rm atext}	\rmfamily
		Sans serif family	\textsf{atext} or {\sf atext}	\sffamily
		`Typewriter family`	\texttt{atext} or {\tt atext}	\ttfamily
2	Series	Medium series (default)	\textmd{atext}	\mdseries
		Boldface series	\textbf{atext} or {\bf atext}	\bfseries
3	Shape	Upright shape (default)	\textup{atext}	\upshape
		Italic shape	\textit{atext} or {\it atext}	\itshape
		Slanted shape	\textsl{atext} or {\sl atext}	\slshape
		CAPS & SMALL CAPS SHAPE	\textsc{atext} or {\sc atext}	\scshape
		Emphasized shape	\emph{atext} or {\em atext}	—
4	Size	Tiny size	{\tiny atext}	\tiny
		Script size	{\scriptsize atext}	\scriptsize
		Foot note size	{\footnotesize atext}	\footnotesize
		Small size	{\small atext}	\small
		Normal size (default)	–	\normalsize
		Large size	{\large atext}	\large
		Larger size	{\Large atext}	\Large
		Largest size	{\LARGE atext}	\LARGE
		Huge size	{\huge atext}	\huge
		Hugest size	{\Huge atext}	\Huge

2. *Font series*: The two series of fonts, medium-valued width and height (default) and boldface, are accessed respectively by the \textmd{} and \textbf{} (or {\bf }) commands (or corresponding \mdseries and \bfseries declarations).
3. *Font shape*: Fonts of four different shapes, upright (default), italic, slanted, and caps and small caps, can be produced respectively through the \textup{}, \textit{} (or {\it }), \textsl{} (or {\sl }) and \textsc{} (or {\sc }) commands (or corresponding \upshape, \itshape, \slshape and \scshape declarations). Apart from these four shapes, fonts of emphasized shape can be produced using the \emph{} or {\em } command.
4. *Font size*: Fonts of ten different sizes can be produced using the {\tiny }, {\scriptsize }, {\footnotesize }, {\small }, {\normalsize }, {\large }, {\Large }, {\LARGE }, {\huge } and {\Huge } commands (or their corresponding declarations of \tiny, \scriptsize, \footnotesize, \small, \normalsize, \large, \Large, \LARGE, \huge and \Huge respectively). The sizes of these ten types of fonts are not rigid, but proportional to the setting made in the document class, e.g., \documentclass[12pt]{} for producing fonts of 12 pt in size.

Notice that italic, emphasized and slanted letters are leaned towards right, for which the gap between the last italic, emphasized, or slanted letter and the following upright letter gets reduced. In order to maintain a proper spacing, the arguments of the {\it }, {\em } and {\sl } commands may be followed by \/. For example, '{\it red} line' will produce '*red* line', while '{\it red\/} line' will produce '*red* line'. The \textit{}, \emph{} and \textsl{} commands make such corrections automatically. Further, the \/ symbol is not required if the last italic, emphasized, or slanted letter is followed by a punctuation. The \/ can also be used between two letters for increasing inter-letter spacing, e.g., 'of\/f\/ice' will produce 'office', while 'office' produces 'office'.

Different combinations of font family, series, shape and size (i.e., the commands of Table 2.1) in a logical way are allowed for producing a wide variety of fonts[2], e.g., \emph{\textbf{emphasized boldface fonts}} for producing '***emphasized boldface fonts***', or {\large\sf large sans serif fonts} for producing 'large sans serif fonts'.

Type of fonts of an individual word or a short phrase can be changed by a font command having an argument, e.g., \textbf{atext} for printing atext in boldface fonts. While an equivalent declaration without any argument, e.g., \bfseries, may be used for changing the fonts of a large portion, say the remaining texts of an environment or a document. To return to the main document fonts, the declaration of any specific font type can be quit using the \normalfont declaration. If a particular font type is to be used in one or more consecutive paragraphs, the font type can be applied as an environment also, e.g., \begin{bfseries}aparas\end{bfseries} for printing aparas in boldface fonts.

2.2 Math-Mode Fonts

Like in text-mode, different types of fonts can be used in math-mode also as shown in Table 2.2 (math-mode is discussed in Hour 11 on page 101). In the case of these math-mode fonts, the following three points are to be noted:

Table 2.2 Different types of math-mode fonts used in LaTeX

Font type	Command	Package required	Output
Serif family	\mathrm{ABC abc}	—	ABCabc
Italic shape	\mathit{ABC abc}	—	ABCabc
Boldface series	\mathbf{ABC abc}	—	$\mathbf{ABCabc}$
Sans serif family	\mathsf{ABC abc}	—	ABCabc
Typewriter family	\mathtt{ABC abc}	—	$\mathtt{ABCabc}$
Mathematical boldface	\boldmath{ABC abc}	amssymb	$\boldsymbol{ABCabc}$
Mathematical normal	\mathnormal{ABC abc}	—	$\mathnormal{ABCabc}$
Calligraphic	\mathcal{A B C}	—	$\mathcal{ABC}$
Open	\Bbb{A B C}	amsfonts/ amssymb	$\mathbb{ABC}$
Open	\mathbb{A B C}	amsfonts/ amssymb	$\mathbb{ABC}$
German/ Fraktur	\mathfrak{ABC abc}	eufrak/ amsfonts/ amssymb	$\mathfrak{ABCabc}$

[2]Different combinations of font family, series, shape and size (i.e., the commands of Table 2.1) in a logical way are allowed for producing a wide variety of fonts.

1. If used in text-mode, the commands of Table 2.2 (except **\boldmath{}**) are to be written within a pair of **$** symbol, e.g., **$\mathbf{**abc**}$** for printing **abc**. In the case of the **\boldmath{}** command, the argument is to be enclosed in a pair of **$** symbol, e.g., **\boldmath{$**abc**$}** for printing ***abc***.
2. The **\mathcal{}**, **\mathbb{}** and **\Bbb{}** commands do not accept lower case letters.
3. Any blank space in the arguments of the commands of Table 2.2 is omitted. Commands, like **\,** or **~**, may be used for maintaining some gap between two letters or words, e.g., **$\mathbb{**A**\,**B**~**C**}$** will produce $\mathbb{A}\,\mathbb{B}~\mathbb{C}$ (creating blank spaces is discussed in §3.6 on page 21, while texts in math-mode in §12.1 on page 113). However, most of the commands of different family, series, and shape having the forms of **\text. . {}** (e.g., **\textbf{}** or **\textit{}**) and **\emph{}**, as shown in Table 2.1, can be used for writing normal texts in math-mode preserving the space provided between two letters in the input file.

2.3 Emphasized Fonts

Important texts in a document are usually emphasized by writing them in **boldface**, *italic*, or in ***boldface italic***, which are done in LaTeX through the **{\bf }**, **{\it }** (or **{\em }**) and **{\bf\em }** commands respectively, or through their other forms of **\textbf{}**, **\textit{}** (or **\emph{}**) and **\textbf{\emph{}}** commands[3] respectively (refer §2.1 for detail).

Apart from the above forms, texts can also be emphasized by underlining them through the **\underline{}** command, e.g., **\underline{**important**}** will produce important. However, the **\underline{}** command does not permit any line break in between its argument, for which it cannot be used for printing a long statement as it may go beyond the margin of a line.

The above problem with the **\underline{}** command can be sorted out using the **ulem** package, which redefines the **{\em }** and **\emph{}** commands for printing their arguments by underlining with required line breaks. If some texts are to be underlined, as well as to be printed in *italic* fonts, either **{\it{\em }}** or **\textit{\emph{}}** command may be used. If the **ulem** package is loaded, the redefined effects of the **{\em }** and **\emph{}** commands can be turned on or off using the **\ULforem** or **\normalem** command in between the texts where from the effects are to be turned on or off respectively. Besides the **{\em }** and **\emph{}** commands, the **ulem** package provides the **\uwave{}**, **\sout{}** and **\xout{}** commands for printing an argument, respectively, by a wavy underline, a strike-out line and by crossing out each character of the argument[4].

---

[3]Both the **\emph{}** and **{\em }** commands produce emphasized fonts, but the **\emph{}** command produces better spacing than the **{\em }** command does.

[4]Struck out or overlapping texts can also be produced by creating a negative horizontal space between two pieces of texts using the **\hspace{}** command with a negative length as its argument, e.g., '~~striked out~~' can be produced by 'striked out**\hspace{**-1.8**cm }**------------'.

Some examples of various features of the **ulem** package are shown in Table 2.3, where different applications are numbered both in LaTeX input and out files for easily identifying the effect of each LaTeX syntax. A blank line is left in the input file before each application for printing it in a new line (creating new lines is discussed in §3.5.1 on page 19), while the texts in a line preceded by a **%** sign are simply commented. The underlining effects of the redefined **{\em }** and **\emph{}** commands are shown in applications 2–5, where the redefined effects are first switched on by putting the **\ULforem** command before application 2 and then they are switched off by putting the **\normalem** command after application 5. Note that the **\uwave{}**, **\sout{}** and **\xout{}** commands are not affected by the **\normalem** and **\ULforem** commands.

Table 2.3 Various forms of emphasized texts under the **ulem** package

LaTeX input	Output
\documentclass[11pt,a4paper]{article} \usepackage{ulem} \begin{document} 1. {\em Normal emphasized texts.} \ULforem % Redefining effects of {\em } and \emph{}. 2. {\em Underlined texts with line breaks.} 3. \emph{Yet underlined texts with line breaks.} 4. {\it{\em Underlined texts in italic fonts …}} 5. \textit{\emph{Yet underlined texts in …}} \normalem % For normal effects of {\em } and \emph{}. 6. {\em Returned to normal emphasized texts.} 7. The next phase is \uwave{wavy underlined.} 8. The next phase is \sout{striked out statement.} 9. The next phase is \xout{crossed out statement.} \end{document}	1. *Normal emphasized texts.* 2. Underlined texts with line breaks. 3. Yet underlined texts with line breaks. 4. *Underlined texts in italic fonts with necessary line breaks.* 5. *Yet underlined texts in italic fonts with necessary line breaks.* 6. *Returned to normal emphasized texts.* 7. The next phase is wavy underlined. 8. The next phase is ~~striked out statement.~~ 9. The next phase is ~~crossed out statement.~~

2.4 Colored Fonts

Like many word-processors, LaTeX also has the provision for producing colored fonts, supported by the **color** package. There are basically three types of color combinations – black and white (**gray**), additive primaries (**rgb**) and subtractive primaries (**cmyk**)[5]. Using the **\definecolor{}{}{}** command, various colors can be defined by setting different values to **gray** and each of the letters of **rgb** and **cmyk** as follows:

\definecolor{`cname`**}{gray}{**w**}** ; $w \in [0, 1]$
\definecolor{`cname`**}{rgb}{**w, x, y**}** ; $w, x, y \in [0, 1]$
\definecolor{`cname`**}{cmyk}{**w, x, y, z**}** ; $w, x, y, z \in [0, 1]$

The definitions of the predefined colors, as well as some examples of defining new colors, are shown in Table 2.4, where `cname` is the name of the user-defined new color. The predefined colors (**black**, **white**, **red**, **green**, **blue**, **cyan**, **magenta** and **yellow**) need not to be redefined, while the user-defined colors (as shown through an example in Table 2.4 under each category of color combination) may be defined in the preamble for global effect or inside the **document** environment for local effect.

Table 2.4 Predefined as well as some user-defined colors for fonts

Type	**Command**	**Color**
Black	\definecolor{black}{gray}{0}	Predefined black
and	\definecolor{white}{gray}{1}	Predefined white
white	\definecolor{cname}{gray}{0.75}	User-defined
	\definecolor{red}{rgb}{1,0,0}	Predefined red
	\definecolor{green}{rgb}{0,1,0}	Predefined green
Additive	\definecolor{blue}{rgb}{0,0,1}	Predefined blue
primaries	\definecolor{black}{rgb}{0,0,0}	Predefined black
	\definecolor{white}{rgb}{1,1,1}	Predefined white
	\definecolor{cname}{rgb}{0,0.7,0.3}	User-defined
	\definecolor{cyan}{cmyk}{1,0,0,0}	Predefined cyan
	\definecolor{magenta}{cmyk}{0,1,0,0}	Predefined magenta
Subtractive	\definecolor{yellow}{cmyk}{0,0,1,0}	Predefined yellow
primaries	\definecolor{black}{cmyk}{1,1,1,1}	Predefined black
	\definecolor{white}{cmyk}{0,0,0,0}	Predefined white
	\definecolor{cname}{cmyk}{0.2,1,0.7,0}	User-defined

Once different colors are defined as above (if required), colored texts can be produced through the **\textcolor{**`cname`**}{**`atext`**}** command, where `atext` is the piece of texts to be colored by `cname` color. For example, **\textcolor{blue}{**`this is in blue`**}** will print '`this is in blue`', while **\textcolor{**`urgb`**}{**`this is in rgb =` **\{**`0,0.7,0.3`**\}}** will print '`this is in rgb = {0,0.7,0.3}`', where `urgb` is a new color defined as **\definecolor{**`urgb`**}{rgb}{**`0,0.7,0.3`**}** (recall that red color is adopted in this book for writing LaTeX commands and other LaTeX syntax).

The fonts discussed in §2.1–§2.3 can also be colored through the **\textcolor{}{}** command, e.g., **\textcolor{magenta}{\small\sf** `small Sans serif in magenta`**}** will produce 'small Sans serif in magenta', and **\textcolor{blue}{$\mathfrak{**`Colored~Fraktur~fonts`**}$}** will produce '$\mathfrak{Colored~Fraktur~fonts}$'.

[5]There are basically three types of color combinations – black and white (**gray**), additive primaries (**rgb**) and subtractive primaries (**cmyk**).

Hour 3

Formatting Texts I

Although LaTeX has numerous predefined macros for automatic and uniform formatting of a document without any mistake[1], many LaTeX users commit mistake by attempting to format a document manually. Manual formatting includes inserting vertical or horizontal spacing with **\bigskip**, **\vskip**, **\vfill**, etc., setting sectional headings via explicit font instructions, such as **\centerline{\bf** `Section 1: Introduction`**}** or **{\bf** `Theorem 4:`**}**, etc. Another mistake may be committed in keeping track of manually set sectional units. Various options for automatic formatting of a document are discussed here.

3.1 Sectional Units

Various sectional units, like chapters and sections, are generated using the **\chapter{}**, **\section{}**, **\subsection{}**, **\subsubsection{}**, **\paragraph{}** and **\subparagraph{}** commands, whose argument is the heading or title of a sectional unit, e.g., the current section of this book is written as **\section{**`Sectional Units`**}**. The sectional unit commands work in order and hence they should be nested properly, i.e., a **\subsection{}** command should follow a **\section{}** command or a **\subparagraph{}** command should follow a **\paragraph{}** command. LaTeX assigns three-tier serial numbers to chapters, sections, subsections, and subsubsections (paragraphs and subparagraphs are not numbered). These numbering system is shown in Fig. 3.1.

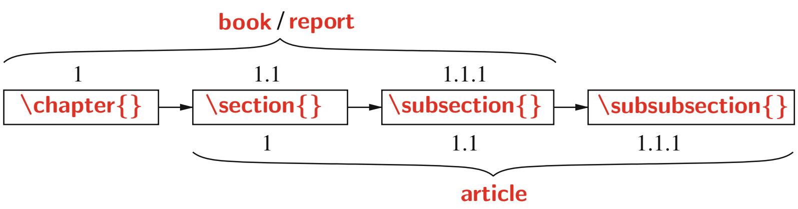


Fig. 3.1 Default three-tier numbering of sectional units

[1] LaTeX has numerous predefined macros for automatic and uniform formatting without any mistake.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_3

In the case of document-class **report** or **book**, which is composed of chapters, numbering is possible only to the **\chapter{}**, **\section{}**, and **\subsection{}** commands. Even if a **\subsubsection{}** command is used, it will not be numbered in the document-class **report** and **book**. A chapter is numbered by a whole number preceded by the label-word '`Chapter`' and followed by its heading. A chapter may contain a number of sections, which are designated as 1.2, 3.5, etc., where the second number is the serial number of a section and the first number is the serial number of the chapter in which the section belongs. Similarly, the third tier of serial numbers are assigned to subsections, which are numbered as 2.5.3, 6.2.2, etc. On the other hand, since the document-class **article** is composed of sections and paragraphs only (it does not support the **\chapter{}** command), the three-tier numbering is assigned to the **\section{}**, **\subsection{}** and **\subsubsection{}** commands, whose numbering is similar with the numbering of **\chapter{}**, **\section{}** and **\subsection{}** commands in the document-class **report** and **book** (however, the first numbering is not preceded by any label-word as done by '`Chapter`' in **report** and **book**). Note that the document-class **letter** does not support any sectional unit.

In some cases, the numbering of a sectional unit may need to be omitted (e.g., the *Preface* of a book is usually written under the **\chapter{}** command, but it is not numbered). LaTeX takes care of it through its **\chapter*{}**, **\section*{}**, **\subsection*{}** and **\subsubsection*{}** commands, which are not numbered because of their starred (*****) forms.

3.2 Labeling and Referring Numbered Items

Like to sectional units addressed in §3.1, LaTeX assigns serial numbers to many environments or elements of an environment (e.g., **table**, **figure**, **equation**, or **\item** as discussed in following Hours). This default numbering system eliminates the possibility of committing any mistake as may happen in manual numbering. Moreover, LaTeX allows to label a numbered item by a unique *reference key*, which can be used to refer the item in any part within the same document (unnumbered items, say **\paragraph{}**, cannot be referred in this way). The labeling and referring of an item are performed through **\label{**`rkey`**}** and **\ref{**`rkey`**}** respectively, where `rkey` is the assigned unique reference key of the item[2].

An example of labeling and referring an item is shown in Table 3.1 on the next page. The **\section{}** commands are followed by the **\label{}** commands containing the reference keys of '`sec:cg`' and '`sec-ex`' respectively (two different types of reference keys are considered to show that labeling can be done in any fashion, but without any blank space in between the texts of a reference key). In the body of the second **\section{}**, the first **\section{}** is referred through the **\ref{}** command using

[2]A numbered item can be labeled using **\label{**`rkey`**}** and then it can be referred using **\ref{**`rkey`**}** anywhere within the same document, where `rkey` is the assigned unique reference key of the item.

Table 3.1 Labeling and referring numbered items

LaTeX input	Output

`\section{Centre of gravity}\label{sec:cg}`
`This is the point though which the resultant of the gravitational forces of all elemental weights of a body acts.`
`%`
`\section{Centre of mass}\label{sec-ex}`
`The definition of the centre of gravity is given in Section~\ref{sec:cg} ...`
	3.2 Centre of gravity
	This is the point though which the resultant of the gravitational forces of all elemental weights of a body acts.
	3.3 Centre of mass
	The definition of the centre of gravity is given in Section 3.2 …

its reference key as the argument of the command. As a result, the first section is referred in the output automatically by its serial number (not by its reference key), thus eliminates any possibility of referring an item by a wrong number.

Similar to the `\ref{}` command, there exists some commands for printing the page number where a numbered item appears. Such commands include `\pageref{rkey}`, `\vref{rkey}` and `\vpageref{rkey}`[3].

- ▷ The `\pageref{}` command prints the page number where its argument is labeled, e.g., '`CG is defined on page~\pageref{sec:cg}`' will print '`CG is defined on page 17`'. The `\pageref{}` command may not be preferred, particularly when labeling and referring of an item appear on the same page, as the command will print the serial number of a page on that page itself as done in the above example. Instead of generating the page number, some other texts, such as '`on this page`' may suit better in this situation. Such facilities are provided in the `\vref{}` and `\vpageref{}` commands, which are defined in the **varioref** package.
- ▷ The `\vref{}` command usually executes a pair of `\ref{}` and `\pageref{}` commands.
  - – It executes only the `\ref{}` command when labeling and referring of an item appear on the same page.
  - – In addition to executing the `\ref{}` command, the `\vref{}` command also generates strings like '`on the facing page`', '`on the preceding page`' or '`on the next page`' when the page numbers of labeling and referring of an item differ by one.
  - – When the difference in page numbers of labeling and referring of an item is more than one, the `\vref{}` command executes both the `\ref{}` and `\pageref{}` commands. For example, '`CG is defined in \S\vref{sec:cg}`' will print '`CG is defined in` §3.2', while '`quoted texts are discussed in \S\vref{sec:quote}`' will print '`quoted texts are discussed in` §3.4 `on the next page`', or '`text-mode fonts are discussed in \S\vref{sec:text-fonts}`' will print '`text-mode fonts are discussed in` §2.1 on page 9'.

[3]The `\pageref{rkey}`, `\vref{rkey}` and `\vpageref{rkey}` commands may be used in a similar way to `\ref{}` for referring the page number of an item, whose assigned unique reference key is `rkey`.

▷ If the \ref{} command is to be ignored, but strings similar to that produced by the \vref{} command are required, the \vpageref{} command may be used. For example, 'CG is defined \vpageref{sec:cg}' will print 'CG is defined on the preceding page'. Two more controls can be obtained through two optional arguments to the \vpageref{} command as \vpageref[asame][adiffer␣]{rkey}, where asame specifies the texts to be printed when labeling and referring appear on the same page, while adiffer specifies the texts if they fall on different pages. For example, 'the \vpageref[above definition][definition␣]{sec:cg} is for the center of gravity' will produce 'the definition on the preceding page is for the center of gravity'. Notice that a blank space (␣) is maintained after adiffer in \vpageref[asame][adiffer␣]{rkey}, i.e., after the string of the second optional argument, for maintaining a gap between the string and the page number (this is not required for the first argument, in which case no page number is generated).

3.3 Texts Alignment

By default LaTeX prints texts with both side aligned. Other types of alignment can be obtained through the **flushleft**, **flushright** and **center** environments, which print texts left, right, and center aligned, respectively. Some applications of these environments are shown in Table 3.2. Such alignments can also be produced using the corresponding declarations of these environments, which are \raggedright, \raggedleft and \centering, respectively.

Table 3.2 User-defined alignments of texts

LaTeX input	Output
\begin{flushleft} \LaTeX\ prints texts with both side aligned. Left aligned texts can be produced through the 'flushleft' environment. \end{flushleft}	LaTeX prints texts with both side aligned. Left aligned texts can be produced through the 'flushleft' environment.
\begin{flushright} \LaTeX\ prints texts with both side aligned. Right aligned texts can ... \end{flushright}	LaTeX prints texts with both side aligned. Right aligned texts can be produced through the 'flushright' environment.
\begin{center} \LaTeX\ prints texts with both side aligned. Center aligned texts can ... \end{center}	LaTeX prints texts with both side aligned. Center aligned texts can be produced through the 'center' environment.

3.4 Quoted Texts

Often some texts are required to be quoted within single or double quotation marks, which are usually done using (') and (") symbols (generally both appear on the same button of a keyboard). However, LaTeX suggests to use (`) as the left-hand single quote and (') as the right-hand single quote (` generally appears in a keyboard on the same button with ∼). For double quotation mark, the single quote symbols may be used twice. For example, `single-quote' will produce ‘single-quote’, while ``double-quote'' will produce “double-quote”.

Sometime an existing statement may need to be quoted without any change, which is usually done in a narrowed width of a page. LaTeX provides the **quote** and **quotation** environments for that purpose. Both the environments are used in the same way, but the **quote** is usually for a short display, while the **quotation** for quoting more than one paragraph. Quoted statements can also be printed in a different line spacing through the **spacing** environment. An application of the **quotation** and **spacing** environments is shown in Table 3.3, where the mandatory argument of the **spacing** environment (1.2 is taken here) specifies the line spacing to be maintained within the

Table 3.3 Quoted texts in a narrowed width and specified line spacing

LaTeX input	**Output**

\LaTeX\ prints texts with both side aligned,
covering the specified width of a page.
\begin{quotation}
\begin{spacing}{1.2}
Quoted statements are also printed with both side
aligned, but in a narrowed width.
    \begin{flushright}
    {\it - Anonymous}
    \end{flushright}
\end{spacing}
\end{quotation}
The 'quotation' environment is used for printing
quoted statements in a narrowed width.
	LaTeX prints texts with both side aligned, covering the specified width of a page.
	
	Quoted statements are also printed with both side aligned, but in a narrowed width.
	
	- Anonymous
	
	The ‘quotation’ environment is used for printing quoted statements in a narrowed width.

environment. If single line spacing is required, instead of the **spacing** environment, the **singlespace** environment may also be used without any argument. Both the **spacing** and **singlespace** environments are supported by the **setspace** package. Notice the nested environments in Table 3.3, where three environments, **quotation**, **spacing** and **flushright**, are nested. It is permitted in LaTeX to create such nested environments.

3.5 New Lines and Paragraphs

LaTeX does not respond to a new line or paragraph set manually by pressing the enter button of the keyboard. Unless specified commands are used, LaTeX considers everything in a single line and single paragraph.

3.5.1 Creating New Lines

The direct command for creating a new line is **\newline**. The texts preceded by a **\newline** command are printed in a new line. A new line can also be created using a line break command (**\linebreak**, ****, ****, or one or more blank lines) at the end of the previous line[4]. Some applications of these commands, using the same sentence, are shown in Table 3.4. To easily distinguish the differences of the commands, the

Table 3.4 Creating new lines

LaTeX input	**Output**

`(1) There are certain key issues to attract investors, which need to be addressed.`
`\newline`
`(2) There are certain key issues to attract investors, which need to be addressed.\linebreak`
`(3) There are certain key issues to attract investors, which need to be addressed.\\`
`(4) There are certain key issues to attract investors, which need to be addressed.\\\\`
`(5) There are certain key issues to attract investors, which need to be addressed.`
 	
`(6) There are certain key issues to attract investors, which need to be addressed.`
	(1) There are certain key issues to attract investors, which need to be addressed.
	(2) There are certain key issues to attract investors, which need to be addressed.
	(3) There are certain key issues to attract investors, which need to be addressed.
	(4) There are certain key issues to attract investors, which need to be addressed.
	
	(5) There are certain key issues to attract investors, which need to be addressed.
	(6) There are certain key issues to attract investors, which need to be addressed.

sentence is numbered in each case. The effects of **\linebreak** in (2), **** in (4), and a blank line prior to (6), are most noticeable. The **\linebreak** command forces the texts of the last line to cover the entire page width. The **** command not only prints the following texts in a new line, but creates a blank line also. On the other hand, a blank line in the input file creates a new line with indentation (some blank space) at the beginning of the line (same effect would be resulted even if more than one blank line were left).

Some extra vertical space above the next new line can be specified in **[]** after the **** command at the end of the previous line, e.g., **\\[2mm]** will create an extra vertical space of 2 mm above the next line. Further, the ***** or ***[]** command may be used for line breaking, but preventing page breaking at that point.

3.5.2 Creating New Paragraphs

Although a new paragraph can be started manually by creating a new line as discussed in §3.5.1, LaTeX suggests to use its predefined macros for creating uniform paragraphs. The direct command for creating a new paragraph is **\par**. On the other hand, the **\paragraph{}** and **\subparagraph{}** commands can also be used for creating new paragraphs with the arguments of the commands as the headings of the paragraphs. Table 3.5 on the facing page shows some applications of these commands, along with three more commands **\parindent**, **\noindent** and **\parskip**. The first **\par** command prints the texts in (1) in a new paragraph with default indentation. The indentation size for paragraph (2) is increased to 8 mm through the **\parindent** = 8**mm** command, while the indentation for paragraph (3) is skipped using the **\noindent**

[4] LaTeX does not create a new line set manually by pressing the `enter` button of the keyboard, rather the same can be obtained by inserting preferably **\newline** or **** at the end of the previous line.

Table 3.5 Creating new paragraphs

LaTeX input	Output

`\par`
`(1) There are certain key issues to attract investors, which need to be addressed.`
`%`
`\par \parindent = 8mm`
`(2) There are certain key issues to attract investors, which need to be addressed.`
`%`
`\par \noindent \parskip 2mm`
`(3) There are certain key issues to attract investors, which need to be addressed.`
`\par`
`(4) There are certain key issues to attract investors, which need to be addressed.`
`%`
`\paragraph{(5) Investing policies:}`
`There are certain key issues to attract investors, which need to be addressed.`
`%`
`\subparagraph{(6) Investing policies:}`
`There are certain key issues to attract investors, which need to be addressed.`
	(1) There are certain key issues to attract investors, which need to be addressed.
	(2) There are certain key issues to attract investors, which need to be addressed.
	
	(3) There are certain key issues to attract investors, which need to be addressed.
	
	(4) There are certain key issues to attract investors, which need to be addressed.
	
	(5) Investing policies: There are certain key issues to attract investors, which need to be addressed.
	
	(6) Investing policies: There are certain key issues to attract investors, which need to be addressed.

command (indentation can be reactivated using the **\indent** command). Further, a vertical space of 2 mm is created above paragraph (3) using the '**\parskip** 2**mm**' command. Note that the effects of the **\parindent** and **\parskip** commands are global, which will act on all the paragraphs preceded by them[5]. These can be viewed in paragraphs (2), (3), and (4). Although the **\noindent** command is used before paragraph (3), paragraph (4) is indented by 8 mm following the **\parindent** = 8**mm** command used before paragraph (2). On the other hand, the **\paragraph{}** command prints its argument as the heading of paragraph (5). Moreover, the **\paragraph{}** command starts paragraph (5) without any indentation and also separates it from paragraph (4) by some predefined vertical blank space. The effect of the **\subparagraph{}** command, shown in paragraph (6), is similar with that of the **\paragraph{}** command, the only difference is that it maintains some indentation at the beginning of the paragraph.

3.6 Creating and Filling Blank Space

Excess blank spaces, created by pressing the `spacebar` or `tab` button of the keyboard, are just ignored in LaTeX, i.e., a sequence of blank spaces is treated as a single one only (similar to converting a sequence of blank lines into a single new line as discussed in §3.5.1). LaTeX provides its own commands for creating a blank space of a specified size, both in horizontal and vertical directions[6], which are given in

[5]The **\parindent** and **\parskip** commands are generally used in the preamble to make their effects globally in all the paragraphs of a document.

[6]Excess blank spaces cannot be created by pressing the `spacebar` or `tab` button of the keyboard, rather LaTeX provides some commands for creating blank spaces of specified sizes both in horizontal and vertical directions.

Table 3.6 Creating blank spaces

Command	Package	Function	Application	
\quad	—	Predefined horizontal single space	x\quad␣y	x y
\qquad	—	Predefined horizontal double space	x\qquad␣y	x y
\, or \thinspace	—	Predefined horizontal thin space	x\,y	x y
\: or \medspace	amsmath	Predefined horizontal medium space	x\:y	x y
\; or \thickspace	amsmath	Predefined horizontal thick space	x\;y	x y
\!	amsmath	Predefined horizontal negative thin space	x\!y	xy
\!\!	amsmath	Predefined horizontal negative medium space	x\!\!y	xy
\!\!\!	amsmath	Predefined horizontal negative thick space	x\!\!\!y	xy
\bigskip	—	Predefined vertical space	refer Table 3.7	
\vskip	—	User defined vertical space	refer Table 3.7	
\vspace{}	—	User defined vertical space	refer Table 3.7	
\hspace{}	—	User defined horizontal space	refer Table 3.7	
\vfill	—	Vertical space to fill up a page	—	
\hfill	—	Horizontal space to fill up a line	refer Table 3.7	

Table 3.6 (commands of the first part are applicable in both text-mode and math-mode). Note in Table 3.6 that there should not be any blank space on any side of \,, \:, \;, \!, \!\!, and \!\!\! commands; while \quad and \qquad commands must be followed by a blank space (a command, ended by an alphabet and followed by another alphabet, must be followed by a blank space in order to avoid the formation of a different command which even may not exist). The need of a blank space after a command, ended by an alphabet and followed by another alphabet, can be avoided by writing the following alphabet or word in {}, e.g., 'x\quad{y}' to produce the same output as that by 'x\quad␣y'. The commands ended by a punctuation can also be applied in this way, e.g., 'x\,{y}' in place of 'x\,y'.

Some applications of the commands given in the second part of Table 3.6 are shown in Table 3.7, where the \bigskip command followed by the line break

Table 3.7 Applications of some blank space creating commands

LaTeX input	Output
\begin{center} \LaTeX\ in 24 Hours\bigskip\\ A Practical Guide for Scientific Writing \end{center}	LaTeX in 24 Hours A Practical Guide for Scientific Writing
\begin{center} \LaTeX\ in 24 Hours \vskip 8mm A Practical Guide for Scientific Writing \end{center}	LaTeX in 24 Hours A Practical Guide for Scientific Writing
\begin{center} \LaTeX\ in 24 Hours \vspace{8mm}\\ A Practical Guide for Scientific Writing \end{center}	LaTeX in 24 Hours A Practical Guide for Scientific Writing
Writing language: \hspace{5mm} English.	Writing language: English.
Marks: 100 \hfill Time: 3 Hours.	Marks: 100 Time: 3 Hours.

command \\ (or any other new line command addressed in §3.5.1) creates a predefined vertical positive space between two lines. In contrary, the \vskip or \vspace{} command creates a user-defined vertical space between two lines or paragraphs. Similarly, the \hspace{} command produces a user-defined horizontal space between two words. Note that the argument of the \vskip command is written with a blank space, while those of the \vspace{} and \hspace{} commands are written in {}. Moreover, the \vspace{} command is to be followed by the line break command \\, otherwise some texts from the next line may be shifted to the previous line to fill up its blank space, if any.

The unit of the rigid lengths in the arguments of the \vskip, \vspace{} and \hspace{} commands can be any one of mm (millimeter), cm (centimeter), in (inch), pt (point), em (width of M) and ex (width of x). The unit em is usually preferred for horizontal lengths, while ex for vertical lengths[7]. Apart from these units, a length can also be taken as a fraction of \textheight (height of texts on a page), \textwidth (width of texts on a page) or \linewidth (width of a column), e.g., 0.2\textheight for a vertical space of 20% of \textheight or 0.3\linewidth for a horizontal space of 30% of \linewidth.

LaTeX removes blank spaces from the beginning and end of a line as well as a page, for which the \hspace{} and \vspace{} commands may not have any effect in such places, i.e., at the beginning and end of a line or a page. If so, the commands may be preceded by a ~. Alternatively, the starred forms of the commands, i.e., the \hspace*{} and \vspace*{} commands, may be used for producing blank spaces at the beginning or end of a line and a page, respectively. Similarly, instead of the \hfill[8] and \vfill commands (abbreviations of the \hspace{\fill} and \vspace{\fill} commands respectively), the \hspace*{\fill} or \vspace*{\fill} command is to be used for filling a line (or a page) by creating necessary horizontal (or vertical) space at the beginning or end of the line (or the page).

Note that a positive valued length argument to the \vskip, \vspace{} or \hspace{} command will create a positive blank space, while a negative valued length argument will reduce space by that amount, e.g., \hspace{-7mm} will reduce the space between two words by 7 mm (words may get overlapped also, refer foot note 4 on page 12). The \vskip or \vspace{} command with a negative value may be used (particularly before and after equations, listed items, tables, figures, etc.) to shrink vertical space enabling to accommodate more amount of materials on a page (§ 5.1.3 on page 39 describes other techniques for the same purpose).

Further, note that the space created by \hspace{} may be limited to the available space in a line even for a larger input value in \hspace{}, while the space created by \hspace*{} with a larger input value (either between two words or at one side of a line) will always continue beyond the page width or even to the next line.

---

[7]LaTeX accepted units for rigid lengths include mm (millimeter), cm (centimeter), in (inch), pt (point), em (width of M) and ex (width of x), where em is usually preferred for horizontal lengths and ex for vertical lengths.

[8]The \dotfill and \hrulefill commands are similar to the \hfill command, except they fill the gap by dots and horizontal lines respectively.

3.7 Producing Dashes Within Texts

LaTeX provides dashes of three different lengths: -, –, and —, which are produced by -, -- and ---, respectively. Out of these dashes, the shortest one is used between inter-related words, the medium one is used to indicate a range, while the longest one is used to show the extension of an expression. Example of each of these three dashes is shown in Table 3.8.

Table 3.8 Dashes of different lengths

LaTeX input	Output
`Inter-related`	Inter-related
`May--August`	May–August
`Weather condition --- like clear sky`	Weather condition — like clear sky

3.8 Preventing Line Break*

Because of the full alignment in a predefined page width, a long statement may need to be split between two words, or even a single word into two parts, continuing the rest in the next line. However, sometime the splitting of a statement between two particular words may need to be prevented, e.g., it is not desirable to split '`Dilip Datta`' into '`Dilip`' and '`Datta`', and then to place them in two lines. Such phrases can be forced to be printed in the same line by using the ~ or \, command between the words without any gap, e.g., '`Dilip`~Datta' or '`10`\,inch'. The \nolinebreak is another command for preventing line breaking at a particular point, e.g., `Dilip`\nolinebreak\␣`Datta`', where ␣ means a blank space (the \nolinebreak command is to be followed by \␣, refer §1.5.1 on page 5 for detail).

The ~, \, and \nolinebreak commands, however, are always not appropriate, particularly when a word is also to be prevented from splitting. If these commands are set manually between two letters of a word, some undesirable gap may get generated within the word. In that case, the appropriate option is to use the \mbox{} command, which prevents its argument from splitting in two lines[9]. For example, 'NSGA-II-UCTO' can be forced to be printed in the same line using '\mbox{`NSGA-II-UCTO`}'. Note that the ~, \,, \nolinebreak and \mbox{} commands may sometime print texts beyond the page width. So, it would be a good practice to check the output after the use of such commands.

3.9 Adjusting Blank Space After a Period Mark*

Generally a sentence is ended by a period (full-stop) mark (.) and the next sentence is started with some prespecified gap after that period mark. Sometime a period mark is to be used within a sentence also, e.g., '`Dr. Datta`', which LaTeX will interpret

[9]The \mbox{} command may be used to prevent its argument from splitting in two lines.

as the end of a sentence and hence will print '`Dr.`' and '`Datta`' with a big gap between them. Such a big gap can be reduced by commands like **\,** and **~** without any blank space on any side (reduction of space is minimum under the **\,** command), or **\␣** (␣ means a blank space). If a period mark is followed by a right parenthesis or a right quote, the **** command is to be used after the parenthesis or the quote, otherwise the combination of **** and parenthesis/quote will be treated as a different command. On the other hand, when a sentence is ended by an uppercase letter, the next sentence is started with a smaller gap. For maintaining a proper gap in this case, the **\@** symbol may be used in between the said uppercase letter and the following period mark. Some applications of these commands are shown in Table 3.9.

Table 3.9 Maintaining proper gap after a period (full-stop) mark

LaTeX input	**Output**
`Dr. Datta`	Dr. Datta
`Dr.\,Datta`	Dr. Datta
`Dr.~Datta`	Dr. Datta
`Dr.\␣Datta`	Dr. Datta
`classes (article, letter, etc.) may be used`	classes (article, letter, etc.) may be used
`classes (article, letter, etc.)\ may be used`	classes (article, letter, etc.) may be used
`made in USA. United …`	made in USA. United …
`made in USA\@. United …`	made in USA. United …

3.10 Hyphenating a Word*

If the last word of a line goes beyond the page width, it is generally full-aligned forcibly by breaking the word into two parts with a hyphen to the first part and then putting the second part in the next line. However, in many cases LaTeX may continue such a word beyond the page width, or break it in an unpleasant place. Proper word-breaking in that case may be specified manually by inserting the **\-** command in between a word for forcibly hyphenating at that point, e.g., 'kind**\-**ness' to print 'kind-' at the end of the current line and 'ness' at the starting of the next line. If LaTeX fails to provide proper hyphenation in a paragraph, particularly in the case of a non-dictionary word, it can be put in the **sloppypar** environment also, e.g., `\begin{sloppypar}Maneswarirabha\end{sloppypar}`. However, it would be a good practice to check the output and subsequently to do some manual adjustment, if required.

Hour 4

Formatting Texts II

The basic formatting of a document is discussed in detail in Hour 3. Some advanced formatting, including default as well as user-defined settings, are discussed in this Hour, like foot note, multiple columns, mini page, marginal note, modified sectional unit, etc.

4.1 Increasing Depth of Sectional Units*

As mentioned in §3.1 on page 15, LaTeX assigns a three-tier serial numbers to the sectional units, which are **\chapter{}**, **\section{}**, and **\subsection{}** in the document-class **book** or **report**, and **\section{}**, **\subsection{}**, and **\subsubsection{}** in the document-class **article** (the document-class **letter** does not support any sectional unit command). Additionally, the **\part{}** command can be used in the document-class **book**, **report**, and **article** for dividing a document into parts. In the document-class **book** or **report**, the levels of **\part{}**, **\chapter{}**, **\section{}**, and **\subsection{}** commands are –1, 0, 1, and 2 respectively, while those of the **\part{}**, **\section{}**, **\subsection{}**, and **\subsubsection{}** commands in the document-class **article** are 0, 1, 2, and 3 respectively. In order to numbering a few more or less levels, the **\addtocounter{}{}** or **\setcounter{secnumdepth}{}** command may be used in the preamble, e.g., **\addtocounter{secnumdepth}**{1} will increase the numbering one level ahead or **\setcounter{secnumdepth}**{5} will extend the numbering up to the **\subparagraph{}** command. Figure 4.1 shows how the numbering up to the **\subparagraph{}** command

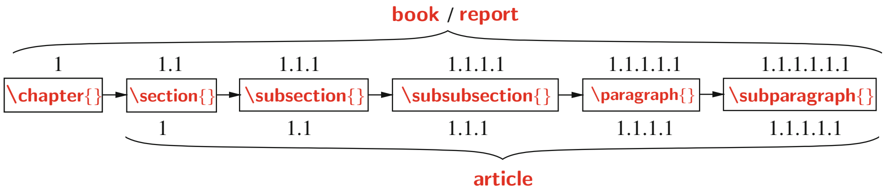


Fig. 4.1 Increasing depth of sectional units

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_4

would be when \setcounter{secnumdepth}{5} is used. If a particular sectional unit is not to be numbered, commands like \chapter*{}, \section*{} or \subsection*{} may be used. On the other hand, if none of the sectional units is to be numbered, instead of using a starred command every time, simply the \setcounter{secnumdepth}{} command may be used with a negative argument, e.g., \setcounter{secnumdepth}{-6}.

4.2 Changing Titles and Counters of Sectional Units*

LaTeX prints sectional units in certain standard patterns. The serial number (in an Arabic numeral) of a chapter is preceded by the label-word 'Chapter', while those of others by none. Some existing commands can be redefined to change these standard patterns. Such an example is shown in Table 4.1. The first \renewcommand{}{}

Table 4.1 Changing titles of sectional units

LaTeX input	\documentclass{book} % \renewcommand{\chaptername}{Unit} \renewcommand{\thechapter}{\Alph{chapter}} \renewcommand{\thesection}{\S~\thechapter.\arabic{section}} \renewcommand{\thesubsection}{\thesection.\arabic{subsection}} % \begin{document} \chapter{Changed Headings} \section{New form of Section} ... \subsection{New form of Sub-section also} ... \end{document}
Output	**Unit A** **Changed Headings** **§ A.1 New form of Section** ... **§ A.1.1 New form of Sub-section also** ...

command in the preamble redefines the \chaptername command to Unit (as done in this book to Hour), i.e., the label-word of a chapter is changed from Chapter to Unit (the argument may be left empty if no label-word is required). The second \renewcommand{}{} command redefines the \thechapter command to \Alph{chapter} to print the serial number of a chapter by an uppercase alphabet[1]. On the other hand, the third \renewcommand{}{} command is used to redefine the \thesection command

[1]The default printing styles of sectional units can be altered by redefining some commands, such as \chaptername, \thechapter, \thesection, \thesubsection, and \thesubsubsection.

to **\S~\thechapter.\arabic{section}**, where the **\thesection** command controls the numbering of sections. The **\S** command makes the serial number of a section to be preceded by the symbol § with a trailing blank space (due to the following **~** symbol). The serial number is generated through the **\thechapter.\arabic{section}** command, where the **\thechapter** command gives the serial number of the chapter under which the section belongs, followed by a full-stop mark (due to **.** symbol), and then the **\arabic{section}** command prints the serial number of the section by an Arabic numeral. The last **\renewcommand{}{}** command redefines the **\thesubsection** command to **\thesection.\arabic{subsection}** for numbering a subsection in an Arabic numeral, preceded by the number of its section (through the **\thesection** command) and a full-stop mark (due to **.** symbol). Similarly, the default numbering style of other sectional units can also be altered. The other available commands for altering the serial numbers of sectional units include **\alph{}** for numbering in lowercase alphabets, and **\Roman{}** and **\roman{}** for numbering, respectively, in upper and lower Roman numerals.

On the other hand, each type of sectional unit is numbered serially starting from unity. This standard numbering can be changed to start from any intermediate integer through the **\setcounter{**`asec`**}{**`n`**}** command, where `asec` is the name of the sectional unit like **chapter** or **section**, and `n` is the starting counter of the sectional unit (default value of `n` is 0), which is incremented by 1 each time the sectional command is used[2]. For example, **\setcounter{chapter}{**`3`**}** will start the chapter numbering from 4.

4.3 Multiple Columns

The document-classes **article**, **book**, and **report** produce a document in single columns under the default option **onecolumn**. Still they permit to use the **twocolumn** option to **\documentclass[]{}** for producing a document in two columns. If required, some portion of a two-column document can also be produced in a single column by writing that portion in **[]** of the **\twocolumn[]** command (detail is in §19.2.4 on page 187). However, **\twocolumn[]** prints each single-column part on a new page even if sufficient blank space is left on the previous page of the document. Therefore, it is not a good practice to mix up both the options in a single document, except the cases as shown in Table 19.10 on page 188, where the single-column mode is used for printing the abstract of an article in a single column.

[2]The default starting of numbering of a sectional unit from unity can be altered to start from the intermediate number `n+1` by inserting **\setcounter{**`asec`**}{**`n`**}**, where `asec` is the name of the sectional unit like **chapter** or **section**.

4.3.1 Multiple Columns Related Parameters

The visual appearance of a multi-column document may be controlled by the \columnsep, \columnseprule and \columnwidth commands. The \columnsep command specifies the gap between two columns, while \columnseprule specifies the width of the vertical line separating two columns. On the other hand, \columnwidth governs the width of a column and its value is automatically calculated from the values of \columnsep and \textwidth.

The default value of \columnseprule is set as zero to make the vertical line invisible, while the value of \columnsep varies with the used document-class and font size (detail is in §5.1.1 on page 37). Their default effects can be altered by assigning them new values in the same way as shown in Table 5.3 on page 40 for page formatting commands. Table 4.2 shows an example of a two-column document, where a vertical line is generated between the columns by assigning a value of 1 mm to \columnseprule.

Table 4.2 Two columns separated by a vertical line through the \columnseprule command

LaTeX input	Output
\documentclass[twocolumn]{article} \columnseprule = 1mm % \begin{document} The standard document-classes permit to print a document either in a ... \end{document}	The standard document-classes permit to print a document either in a single-column or in a two-column mode. This is an example of a two-column mode where the columns are separated by a vertical line.

4.3.2 A Flexible Approach to Generate Multiple Columns

The standard LaTeX can produce a document in a maximum of two columns through the twocolumn option to the \documentclass[]{} command. Moreover, although the \twocolumn[] command can produce a portion of a two-column document in a single column, each single column is printed on a separate page. On the other hand, the columns on the last page are not balanced. To overcome such problems, the multicols environment, defined in the multicol package, can be used, which has the capability of producing any number of columns (up to ten) even on a single-column page[3]. Moreover, it can be applied inside other environments, like minipage (detail is in §4.4 on the following page). Table 4.3 on the facing page shows an application

[3]The multicols environment, defined in the multicol package, can be used for producing any number of columns (up to ten) even on a single-column page.

Table 4.3 Multiple columns generated through the **multicols** environment

LaTeX input	**Output**
\documentclass[11pt,a4paper]{article} \usepackage{multicol} \columnseprule = 0.5mm % \begin{document} This is an example where a ... % \begin{multicols}{3} This is a three-column paragraph. ... \end{multicols} % In this example, all the three ... \end{document}	This is an example where a paragraph is written in three columns. The paragraphs above and below it are written in single columns. This is a three-column para-graph. The gap between two columns are filled by a vertical line of 0.5 mm width. In this example, all the three paragraphs are pro-breakduced on the same page.

of the **multicols** environment, where three columns on a single-column document are generated through the mandatory argument of the environment. Moreover, two adjacent columns are separated by a vertical line of 0.5 mm width through **\columnseprule**. Because of the default full alignment of texts, excess blank space can be seen in the columns in Table 4.3. In order to avoid such excess blank space, texts may be made left aligned using **\raggedright**, which is to be inserted just after the **\begin{multicols}{**3**}** command.

In Table 4.3, the multi-column portion was shown preceded by a single-column paragraph. Instead of inserting in that way, it can also be inserted as an optional argument to the **multicols** environment as **\begin{multicols}{**3**}[**apref**]**, where apref is the piece of texts to precede the multi-column portion.

The columns under the **multicols** environment can be customized through **\columnsep** and **\multicolsep**. The **\columnsep** command specifies the gap between two columns, while **\multicolsep** controls the vertical gap before and after the environment. The default values of **\columnsep** and **\multicolsep** in this environment are around 3.5 mm and 4.5 mm, respectively. The commands can be assigned other values in the same way as shown in Table 5.3 on page 40 for page formatting commands. On the other hand, the width of a column is calculated from the number of columns to be generated and the current value of **\linewidth**. The **\linewidth** command specifies the width of the current lines of texts, whose value is usually the same with that of **\textwidth** (detail is in §5.1.2 on page 38), but may vary inside some environments like **quotation** (refer §3.4 on page 18 for detail).

The **table** and **figure** environments (details are in Hours 7 and 9) may not be supported properly in the **multicols** environment. In that case, their starred forms, i.e., **table*** and **figure*** environments, may be used for drawing tables or figures spanning all the columns.

4.4 Mini Pages

Mini pages mean dividing a portion of a page into a number of width-wise parts, usually for presenting related materials side-by-side, e.g., two tables for comparison purpose, a picture and its description, or LaTeX input and its output. Mini pages are generated through the **minipage** environment with a pair of optional and mandatory arguments, i.e., `\begin{minipage}[avert]{ahorz}`, where `avert` is the vertical alignment of the mini page and `ahorz` is its horizontal width[4]. The allowed vertical alignment options are center (default), top, and bottom, which are denoted by **c**, **t**, and **b**, respectively. In the case of the horizontal width, the sum of widths of all the side-by-side mini pages should not exceed the line width of the main page, otherwise the excess mini pages will be placed in the following lines. Hence, instead of specifying any fixed numeric value (say, `5cm`), the width of a mini page may be specified as a fraction of the **\linewidth** command, say '`0.4\linewidth`' to create a mini page of width equal to 40 % of the line width of the main page.

Similar to the **minipage** environment, there exits **boxedminipage** environment also, which is defined in the **boxedminipage** package. The basic difference between the environments is that each mini page created by the **boxedminipage** environment is placed in a box. Applications of the environments are shown in Table 4.4, where

Table 4.4 Dividing a page width-wise using the **minipage** and **boxedminipage** environments

LaTeX input	Output

`\documentclass[11pt,a4paper]{article}`
`\usepackage{boxedminipage}`
`%`
`\begin{document}`
`%`
`\begin{minipage}[t]{5.0cm}`
`This minipage is ... width is 5.0\,cm.`
`\end{minipage}`
`%`
`\begin{boxedminipage}[t]{0.4\linewidth}`
`This boxed-minipage is also ...`
`\end{boxedminipage}`
`%`
`\end{document}`
	This minipage is top-aligned and its horizontal width is 5.0 cm. | This boxed-minipage is also top-aligned, but its horizontal width is equal to 40% of the page line-width.

both the environments are top-aligned with **minipage** of 5.0 cm width and that of **boxedminipage** is equal to 40 % of the line width of the main page.

4.5 Foot Notes

LaTeX provides the **\footnote{}** command for printing its argument as a foot note. The command is to be inserted just after the word or phrase (usually without any gap) against which a foot note is to be generated. In the output, such words or phrases are superscribed in Arabic numerals with the counter of the **\footnote{}** command. The corresponding foot notes are printed serially at the bottom of the pages and

[4] The **minipage** environment can be used for dividing a portion of a page into a number of width-wise parts, usually for presenting related materials side-by-side, e.g., two tables for comparison purpose, or a picture and its description.

they are marked by superscribing the respective serial numbers on the left side. As shown in Table 4.5, the foot notes on a page are separated from its main materials

Table 4.5 Foot notes generated through the **\footnote{}** command

LaTeX input	Output
`Both Rubi and Lila\footnote{They are sisters.} study in class I, while Ravi and Joy\footnote{They are friends.\label{fn:friends}} study in class II.`	Both Rubi and Lila[1] study in class I, while Ravi and Joy[2] study in class II. ——— [1]They are sisters. [2]They are friends.

by a small horizontal line. A foot note may contain anything, such as paragraphs, lists, mathematical expressions, tabular materials, etc. Generally a word/phrase and the foot note generated against it appear on the same page. However, a foot note may be continued on the following pages also if it is long enough. On the other hand, the foot notes of a multi-column document are printed in full width at the bottom of a page, irrespective of the columns where from they are generated.

A foot note can be labeled and referred like other numbered items. As shown in Table 4.5, a foot note is to be labeled using **\label{}** inside **{}** of the **\footnote{}** command, and it can be referred as usual by the **\ref{}** command[5].

4.5.1 Foot Notes in Mini Pages*

The markings and positions of foot notes inside a **minipage** environment are quite different. The foot notes of each mini page are marked by lowercase alphabets (a, *b*, c, etc.) and placed them at the bottom of the mini page. An example of foot notes in mini pages, along with a foot note in the main page also, is shown in Table 4.6 on the following page. The line break command **\\[2mm]** is used before the first and after the last **minipage** environments for producing the mini pages in a separate paragraph with a gap of 2 mm before and after the paragraph. Moreover, the **\hfill** command is used after the first **minipage** environment for placing the mini pages left and right aligned with the remaining space of the line width of the main page in between the mini pages. As seen in Table 4.6, the foot note in the main page and those inside the mini pages are counted separately (by Arabic numerals in the main page, while by lowercase alphabets in the mini pages).

[5]Like other numbered items, a foot note generated through **\footnote{}** can also be labeled and referred using **\label{}** and **\ref{}**, respectively.

Table 4.6 Foot notes in mini pages

LaTeX input	**Output**

`The following is an example of … mini`
`page\footnote{Foot note style in mini pages`
`is different.}.\\[2mm]`
`%`
`\begin{minipage}[t]{0.45\linewidth}`
`Both Rubi and Lila\footnote{They are`
`sisters.} study … Joy\footnote{They are`
`friends.} study in class II.`
`\end{minipage}\hfill`
`%`
`\begin{minipage}[t]{0.45\linewidth}`
`Though the milk of a cow\footnote{Domestic`
`animal.} is … milk of a tiger\footnote{Wild`
`animal.}.`
`\end{minipage}\\[2mm]`
`%`
`Foot notes in a mini page are marked by`
`lowercase alphabets and placed …`
	The following is an example of foot notes inside a mini page[1].
	
	Both Rubi and Lila[a] study in class I, while Ravi and Joy[b] study in class II.
	[a] They are sisters.
	[b] They are friends.
	
	Though the milk of a cow[a] is used in various food products, nobody bothers about the milk of a tiger[b].
	[a] Domestic animal.
	[b] Wild animal.
	
	Foot notes in a mini page are marked by lowercase alphabets and placed at the bottom of the mini page.
	
	[1] Foot note style in mini pages is different.

4.5.2 *Altering the Pattern of Foot Notes**

By default, foot notes are numbered in Arabic numerals. Other markings can be obtained by redefining the **\thefootnote** command in the preamble. For example, the **\renewcommand{\thefootnote}{\fnsymbol{footnote}}** command may be used for marking the foot notes by symbols like $*$, †, ‡, §, etc.

Although the markings of foot notes inside mini pages can also be altered by changing their counter **mpfootnote** by redefining the **\thempfootnote** command, still the markings are started afresh in each mini page. Such markings may be confusing in many cases, particularly if a foot note is to be referred. LaTeX provides the **\footnotemark** and **\footnotetext{}** commands through which the foot note of a mini page can also be made to work like foot notes in main pages[6]. That is, the commands alter the counter of the foot notes of a mini page from **mpfootnote** to **footnote**, which is the counter of foot notes of main pages. The application of the **\footnotemark** and **\footnotetext{}** commands is shown in Table 4.7 on the next page. Like the **\footnote{}** command in main pages, the **\footnotemark** command in a **minipage** environment is used just after the word or phrase against which a foot note is to be generated. Then the contents of the foot note are written as the argument of the **\footnotetext{}** command immediate after the **minipage** environment. A foot note of a mini page, generated through the **\footnotemark** and **\footnotetext{}** commands, can be labeled and referred as an ordinary foot note. In that case, as shown in the second **minipage** environment in Table 4.7, the foot note is to be labeled inside **{}** of the **\footnotetext{}** command. However, this process for generating foot notes in mini pages has the drawback that only one foot note can be generated against a **minipage** environment.

[6] LaTeX provides **\footnotemark** and **\footnotetext{}** for generating foot notes in mini pages, similar to those in main pages generated through **\footnote{}**.

Table 4.7 Foot notes of a mini page to work like those in main pages

LaTeX input	Output

`The following is an example of ... mini`
`page\footnote{Foot note style in mini pages`
`is different.}.\\[2mm]`
`%`
`\begin{minipage}[t]{0.45\linewidth}`
`Both Rubi and Lila\footnotemark\␣study in`
`class I.`
`\end{minipage}`
`\footnotetext{They are sisters.}\hfill`
`%`
`\begin{minipage}[t]{0.45\linewidth}`
`The milk of a cow\footnotemark\␣is ...`
`\end{minipage}`
`\footnotetext{Domestic`
`animal.\label{fn:cow}}\\{2mm}`
`%`
`Foot notes in a mini page are marked ...`
	The following is an example of foot notes inside a mini page[1].
	
	Both Rubi and Lila[2] study in class I. — The milk of a cow[3] is used in various food products.
	
	Foot notes in a mini page are marked by lowercase alphabets and placed at the bottom of the mini page.
	
	[1]Foot note style in mini pages is different.
	[2]They are sisters.
	[3]Domestic animal.

The inclusion of the **fnpara** package prints all the foot notes of a page in a single paragraph, instead of stacking them one below another. To put all the foot notes in the right column in a two-column document, the **ftnright** package may be loaded.

Similar to the **\footnote{}** command, the **\endnote{}** command under the **endnotes** package may be used for printing notes at the end of a chapter or document under the heading '`Notes`'. Foot notes can also be printed as end notes by redefining the **\footnote** command as **\renewcommand{\footnote}{\endnote}**.

4.6 Marginal Notes*

LaTeX provides the **\marginpar{}** command for printing an important note in the margin of a document. An application of the command can be seen just outside of this line, which is produced by inserting '**\marginpar{\em** `Marginal note.`**}**' after the word `line` in the current sentence of this book. A marginal note is started against the line of a page in which the **\marginpar{}** command appears. *Marginal note.*

The **\marginpar{}** command usually prints its argument as a marginal note in the right margin of a page. In the case of a **twoside** document, such as **book**, a marginal note is printed in the right margin of an odd numbered page and in the left margin of an even numbered page. In a **twocolumn** document, a marginal note is printed in the nearest margin. Although LaTeX decides the correct margin on its own without bothering a user, a problem may arise if an arrow is to be used in the margin to point some texts. This is because, as shown in the margin of this line, a left-pointing arrow is required in the right margin and a right-pointing one is required in the left margin. However, it is not known beforehand in which margin the arrow will appear, for which it would be confusing whether a right-pointing or a left-pointing arrow is $\Longleftarrow$

to be used. LaTeX avoids this confusion also by allowing the **\marginpar{}** command to take an optional argument, which is processed if the marginal note is to appear in the left margin, otherwise the mandatory argument is processed to print the marginal $\Longrightarrow$ note in the right margin. For example, the arrow in the margin of this line is obtained through the **\marginpar[$\Longrightarrow$]{$\Longleftarrow$}** command.

When the **\marginpar{}** command is used for printing a marginal note, the page size of a document is to be adjusted to accommodate the marginal notes. The commands required in this regard include **\marginparwidth**, **\marginparsep**, and **\marginparpush**. These commands signify, respectively, the width of a marginal note, the horizontal gap between the main texts of a page and a marginal note, and the vertical gap between two successive marginal notes. As addressed in §5.1.2 and shown in Table 5.3 on page 40, all of these three commands are to be put in the preamble, with appropriate values, along with other page setting commands. As an example, in this book, the **\marginparwidth**, **\marginparsep**, and **\marginparpush** commands are assigned the values of 12 mm, 3 mm, and 15 mm, respectively.

Hour 5

Page Layout and Style

LaTeX has default page layouts and styles which can be used without any difficulty. These default settings can also be customized, if required, which are discussed in this Hour.

5.1 Page Layout

The type of the physical paper for producing a document can be specified as an option to the **\documentclass[ ]{ }** command, e.g., **\documentclass[a4paper]{article}** for printing an article on A4-size paper (**a4paper** is the type of paper). The types and sizes of the standard papers accepted by the **\documentclass[ ]{ }** command are listed in Table 5.1. The length of a line and the number of lines per page for printing texts are fixed according to the chosen paper-size. Besides the size of a paper, its orientation can also be specified by **portrait** (default) or **landscape** as another option to **\documentclass[ ]{ }**. In the **portrait** orientation, the longer dimension of a page goes in vertical and shorter dimension in horizontal, while the **landscape** orientation is just opposite to the **portrait** orientation.

Table 5.1 Types and sizes of standard papers accepted by the **\documentclass[]{ }** command

Type of paper	Size	Type of paper	Size
a4paper	210 mm×297 mm	**letterpaper** (default)	216 mm×279 mm
a5paper	148 mm×210 mm	**legalpaper**	216 mm×356 mm
b5paper	176 mm×250 mm	**executivepaper**	184 mm×267 mm

5.1.1 Standard Page Layout

There are many LaTeX commands controlling various dimensional parameters of the page layout for a document. The commands controlling some important parameters are listed in Table 5.2 on the next page, as well as shown diagrammatically

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_5

Table 5.2 Commands controlling some important parameters of a page layout

Layout	Command	Function
Main texts	\textheight	Height of main texts without header and footer.
	\textwidth	Width of main texts without marginal notes.
	\oddsidemargin	Blank space on the left margin of odd-numbered pages, if both-side printing is opted (both-side printing can be opted using the **twoside** option to the \documentclass[ ]{ } command).
	\evensidemargin	Blank space on the left margin of even-numbered pages (active only when both-side printing is opted).
Columns	\columnsep	Gap between two columns in multi-column mode.
	\columnseprule	Width of the vertical line separating two columns in multi-column mode (default value is zero so as to make the line invisible).
	\columnwidth	Width of a column in multi-column mode (calculated using the values of \textwidth and \columnsep).
	\linewidth	Width of the lines of texts (usually equal to \columnwidth, but may vary in some environments, like the **quotation** environment).
Header	\headheight	Height of the header.
	\headsep	Vertical gap between the header and the first line of the main texts.
	\topmargin	Extra vertical space above the header.
Footer	\footskip	Vertical gap between the last line of the main texts and the footer.
Marginal notes	\marginparwidth	Width of marginal notes.
	\marginparsep	Horizontal gap between the main texts and marginal notes.
	\marginparpush	Vertical space between two successive marginal notes.

in Fig. 5.1 on page 39. The **\hoffset** and **\voffset** commands in Fig. 5.1 represent, respectively, the horizontal and vertical coordinates of the reference point, whose default values are usually set to zero. All the standard papers listed in Table 5.1 have some fixed values for the commands controlling the dimensional parameters of a page layout. Even for the same paper, the value of a command may vary with the size of fonts as well as with the type of printing (single-side or both-side). However, the values of these commands are independent of the three standard document-classes of **article**, **book**, and **report**. For a particular setting, the page layout similar to the one shown in Fig. 5.1, along with the values of some parameter controlling commands, can be obtained through the **\layout** command defined in the **layout** package. Such values for the document-class **article** in `12`**pt** fonts on **oneside** printing **a4paper** paper are also shown in Fig. 5.1 (obtained just by executing the **\layout** command), where the values of the commands are given in `pt` (1 pt ≈ 0.3515 mm). The commands **\paperheight** and **\paperwidth** represent, respectively, the height and width of the physical paper.

*5.1.2 Formatting Page Layout**

Instead of using a standard page layout, a user can create an own layout by assigning suitable values to the layout controlling commands. Such a command can be assigned a new value either through an explicit mathematical expression or through the **\setlength{**`acomm`**}{**`aval`**}** command, where `acomm` is a command and `aval` is its new value. If a command is not assigned a new value, its default value for the

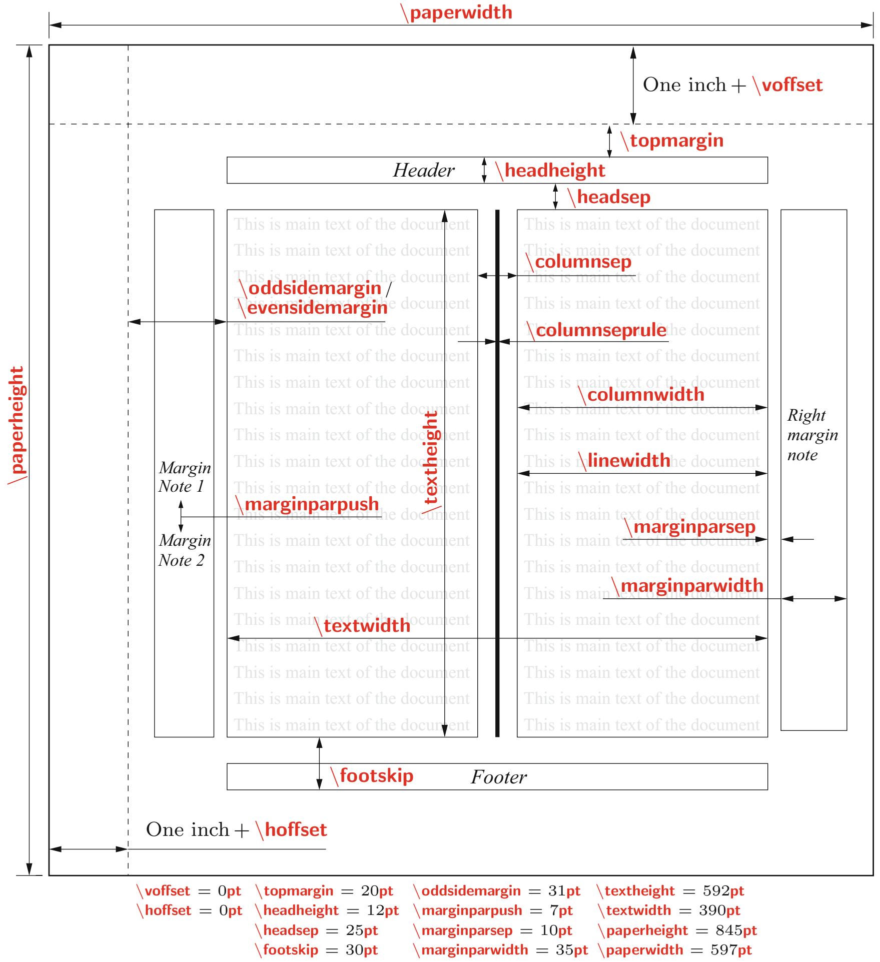


Fig. 5.1 Commands controlling some important parameters of a page layout

chosen paper is used. The values used during drafting this book (the paper is **a4paper**) are shown in Table 5.3 on the next page through both mathematical expression and the **\setlength{}{}** command (any one of these forms is to be inserted in the preamble of the input file of a document).

*5.1.3 Increasing the Height of a Page**

The commands discussed in §5.1.1 and §5.1.2 format a page layout globally, i.e., their values remain the same on all pages. Sometime, however, the vertical height of a particular page may need to be enlarged locally, specially to accommodate a table

Table 5.3 A manually defined page layout (only one form is to be used in the preamble)

Mathematical expression	Use of the \setlength{}{} command
\topmargin = 0mm	\setlength{\topmargin}{0mm}
\oddsidemargin = 15mm	\setlength{\oddsidemargin}{15mm}
\evensidemargin = 0mm	\setlength{\evensidemargin}{0mm}
\textheight = 210mm	\setlength{\textheight}{210mm}
\textwidth = 150mm	\setlength{\textwidth}{150mm}
\marginparwidth = 15mm	\setlength{\marginparwidth}{15mm}
\marginparsep = 3mm	\setlength{\marginparsep}{3mm}
\marginparpush = 7mm	\setlength{\marginparpush}{7mm}
\parindent = 5mm	\setlength{\parindent}{5mm}
\footskip = 10mm	\setlength{\footskip}{10mm}

or a figure, or even small piece of texts of a section going to a new page. This can be done through the \enlargethispage{asize} or \enlargethispage*{asize} command, where \enlargethispage{} increases the vertical height of a page by an amount of asize, while \enlargethispage*{} tries to shrink inter-line spacing to free that amount of space on the page. The commands may be inserted in any place in the input file containing the materials to be printed on the particular page.

5.2 Page Style

Once a page layout is set, the next step is the design of the pages of a document, which mainly includes running header and footer, page numbering, and the front page of a chapter. LaTeX provides various standard macros as well as customizing facilities for designing a document. The header and footer, and page numbering are discussed in the following two sections, while the front page of a chapter is covered in §4.2 on page 28.

5.3 Running Header and Footer

The running header and footer on the pages of a document are controlled by the \pagestyle{} and \thispagestyle{} commands, whose arguments specify a page style. Some commonly used page styles are listed in Table 5.4 on the following page. The \pagestyle{} command implements the chosen page style on the current page as well as on the succeeding pages, while \thispagestyle{} works locally on the current page only. Usually \pagestyle{} is used in the preamble for its global effect, and \thispagestyle{} is used inside the document environment to suppress the global effect of \pagestyle{} on the current page, e.g., \thispagestyle{empty} may be used in the title page of a book or a report.

By default, \thispagestyle{plain} is issued by the document-classes of article, book, and report to the \maketitle command and the first page of major sectioning commands,

Table 5.4 Page styles to control running header and footer in a document

Page style	Package	Function	Application
empty	—	Both the head and foot are empty, and there is no page number (however, the pages are counted).	—
plain	—	The head is empty, but the foot contains the page number at the center of the foot (default in the document-classes **article** and **report**).	—
headings	—	The foot is empty, but the head contains the page number as well as title information as determined by a document-class, like chapter and section headings (does not apply to the first page of a chapter).	Refer §5.3.1
myheadings	—	Same with **headings** except that the page titles in the head are not automatic, but a user has to supply through some commands.	Refer §5.3.2
fancy	**fancyheadings**	Fully user-defined headers and footers.	Refer §5.3.3
fancy	**fancyhdr**	Fully user-defined headers and footers.	Refer §5.3.4

like **\part{}** or **\chapter{}**. To suppress the effect of **\thispagestyle{plain}** on these pages, **\thispagestyle{}** with an appropriate page style may be used just after the **\maketitle**, **\part{}** or **\chapter{}** command, e.g., **\thispagestyle{empty}** may be used to suppress the default page numbering on these pages.

5.3.1 *Header with the* **headings** *Style*

In the page style **headings**, the running footer is empty, and the header contains the page number and the title of a sectional unit of that page. The position of the page number depends on the type of printing, and the sectional unit whose title appears in the heading depends on the chosen document-class. Different types of headings under the page style **headings** are outlined in Table 5.5. In the case of multiple sectional units on a page, the title of the last unit appears on the header. In the heading, the title of a **\chapter{}** is also preceded by the label-word `CHAPTER` and its serial number, while the title of a **\section{}** or a **\subsection{}** is preceded by its serial number only.

Table 5.5 Styles of headers under the **headings** page style

Option	Page	Document-class		
		article	**book**	**report**
oneside	Odd	Title of **\section{}** on left and page number on right	Title of **\section{}** on left, if any, and page number on right	Title of **\chapter{}** on left and page number on right
(One-side printing)	Even	Same with an odd page	Title of **\chapter{}** on right and page number on left	Same with an odd page
twoside	Odd	Title of **\subsection{}** on left, if any, and page number on right	Same with an odd page of one-side printing	Title of **\section{}** on left, if any, and page number on right
(Two-side printing)	Even	Title of **\section{}** on right and page number on left	Same with an even page of one-side printing	Title of **\chapter{}** on right and page number on left

5.3.2 Header with the myheadings Style

The running header in the page style **myheadings** is the same with that in the page style **headings**. The only difference is that the style of the header under **headings** is predefined, while it is user-defined under **myheadings**. The information about a sectional unit is passed to the running header through a specified marker command, e.g., the marker commands associated with **\chapter{}**, **\section{}**, and **\subsection{}** are **\chaptermark**, **\sectionmark**, and **\subsectionmark**, respectively. Further, the **\markboth{}{}** and **\markright{}** commands are associated with a marker command. The **\markboth{**`aeven`**}{**`aodd`**}** command is active under the **twoside** printing option with `aeven` as the contents of the header of even (left hand) pages and `aodd` as that of odd (right hand) pages. In **\markboth{}{}**, page number is printed on left side of even pages and on right side of odd pages. On the other hand, **\markright{**`acont`**}** is appropriate for **oneside** printing option, where `acont` is the contents of the header of all pages.

Generally the information of two different sectional units (e.g., **\chapter{}** and **\section{}**, or **\section{}** and **\subsection{}**) is put on odd and even pages. However, the information of both the sectional units cannot be passed through **\markboth{}{}** associated with the marker command of a single sectional unit. Therefore, **\markboth{**`aeven`**}{}** and **\markright {**`aodd`**}** are used in combination, where `aeven` is the information for left hand pages and `aodd` is that for right hand pages (the second argument of **\markboth{}{}** is not processed). In this combination, `aeven` and `aodd` are internally stored as **\leftmark** and **\rightmark**, respectively.

Once **\markboth{}{}** and **\markright{}** are finalized, they are associated with the marker commands of two different sectional units, where the marker commands are redefined through the **\renewcommand{}[]{}** command to implement the changes made. Two examples of redefining the marker commands **\chaptermark**, **\sectionmark**, and **\subsectionmark** are shown in Table 5.6 (each **\renewcommand{}[]{}**

Table 5.6 Redefining running header generating marker commands

Example	Redefinition of marker commands	Document-class
1	\renewcommand{\chaptermark}[1]% {\markboth{\chaptername~\thechapter. #1}{ }} \renewcommand{\sectionmark}[1]% {\markright{\thesection. #1}}	**book** / **report** with **twoside** printing option
2	\renewcommand{\sectionmark}[1]% {\markboth{\thesection. #1}{ }} \renewcommand{\subsectionmark}[1]% {\markright{\thesubsection. #1}}	**article**

is split into two lines as **\renewcommand{}[]%** and **{}** due to space limit). The **\chaptername** command generates the label-word `Chapter`, while **\thechapter**, **\thesection**, and **\thesubsection** generate the serial numbers of the current chapter, section and subsection, respectively. The syntax **[**`1`**]** means that there is one argument and the syntax **#**`1` is the location where the argument would be printed (in the current examples, it might be the title of a sectional unit). The first \renewcommand{}[]{} command in example 1 would generate a running header on left hand pages something like ‘Chapter 1. Headers and Footers’, while the second \renewcommand{}[]{} command of this example would generate a header on right hand pages something like ‘1.2. Generating Headers’. The redefinitions given in Table 5.6 are default headers under the page style headings discussed in §5.3.1. To get the same headers under the page style myheadings, these redefinitions are to be put in the preamble of the input file of a document. Since the headers under the page style myheadings are user-defined, it is not necessary to use the exact redefinitions given in Table 5.6, but these can be customized as one wishes. For example, the \chaptername command in example 1 may be omitted if the label-word Chapter is not to be printed in the header, or the headers can be produced in boldface fonts by writing the arguments of \markboth{}{} and \markright{} through \textbf{}, like \markboth{\textbf{\thechapter. #1}}{} and \markright{\textbf{\thesection. #1}}.

5.3.3 *Header and Footer with the fancy Style Under the fancyheadings Package**

The page style fancy, defined in the fancyheadings package, allows very elegant customization of the running header and footer of a document. The package provides three types of headers as well as footers, through which a header/footer can be made page-wise left, center or right aligned, or even multiple pieces of headers and footers can be used. The commands for such headers and footers are shown in Table 5.7, where podd and peven are the contents of the headers/footers on odd and

Table 5.7 Commands for headers and footers under the fancyheadings package

Commands for headers	Commands for footers	Alignment
\lhead[peven]{podd}	\lfoot[peven]{podd}	Left
\chead[peven]{podd}	\cfoot[peven]{podd}	Center
\rhead[peven]{podd}	\rfoot[peven]{podd}	Right

even pages, respectively (optional peven becomes active only if twoside printing is opted). The line break command \\ can also be used to put a header/footer in multiple lines. However, in that case the values of \headheight and \footskip may need to be increased (refer Fig. 5.1 and Table 5.3).

In the fancyheadings package, provision is also there for drawing horizontal rules (lines) below the header and above the footer, whose thicknesses are controlled through the \headrulewidth and \footrulewidth commands, respectively. The default value of \headrulewidth is 0.4 pt, while that of \footrulewidth is 0 pt (making the rule above the footer invisible). The thickness of a rule can be altered through \setlength{}{} (refer Table 5.3 for detail). Moreover, the headers, footers and their rules can also be extended to cover the marginal notes, by increasing the value of their width controlling command **\headwidth** from **\textwidth** (default value) to **\textwidth**+**\marginparsep**+**\marginparwidth**.

The first three pages of a book chapter (the front page, an even page, and an odd page), designed through the page style **fancy** under the **fancyheadings** package, are shown in Table 5.8, where the line numbers in the LaTeX input file are shown for explanation purpose only. Two-side printing is opted in line 1 through the **twoside** option to **\documentclass[]{}**. The **fancyheadings** package is loaded in line 2 and the page style **fancy** is opted in line 4. The marker commands **\chaptermark** and **\sectionamark** are redefined in lines 5 and 6 through **\markboth{}{}** and **\markright{}**, respectively. The first argument of **\markboth{}{}** and the only argument of **\markright{}**, which in the

Table 5.8 Header and footer with the **fancy** page style under the **fancyheadings** package

LaTeX input

```
1 \documentclass[12pt,a4paper,twoside]{book}
2 \usepackage{fancyheadings}
3 %
4 \pagestyle{fancy}
5 \renewcommand{\chaptermark}[1]{\markboth{\thechapter. #1}{}}
6 \renewcommand{\sectionmark}[1]{\markright{\thesection. #1}}
7 \lhead[\textbf{\thepage}]{\textbf{\rightmark}}
8 \rhead[\textbf{\leftmark}]{\textbf{\thepage}}
9 \lfoot[\textbf{Engineering Mechanics}]{}
10 \rfoot[]{\textbf{Dilip Datta}}
11 \cfoot[]{}
12 \renewcommand{\headrulewidth}{0.15mm}
13 \renewcommand{\footrulewidth}{0.15mm}
14 \addtolength{\headwidth}{\marginparsep}
15 \addtolength{\headwidth}{\marginparwidth}
16 %
17 \begin{document}
18 ...
19 \chapter{Distributed Force System}
20 \thispagestyle{empty}
21 For simplifying an analysis, the force exerted by a body...
22 ...
23 \section{Centre of Gravity}
24 Since the weight of a body is a system of concurrent forces...
25 ...
26 \end{document}
```

Output

Chapter 4

Distributed Force System

For simplifying an analysis, the force exerted by a body upon another body is usually treated as a point force acting at the point of contact of the bodies. In real sense, no force acts at a point, but over a certain length, area or volume. This may be prominent in the case of the force exerted on the pavement by a wheel of an automobile, particularly when no air is filled in the wheel. In this example, the force is distributed over the area of contact between the wheel and the pavement. A force distributed over a line can be found in a rope-way. A good example of a force distributed over a volume is the weight (gravitational force) of a body. The elemental gravitational force of each element of a body acts toward the centre of the earth, i.e. the gravitational force of the entire body is a system of concurrent forces with the centre of the earth as the point of concurrency.

46 4. Distributed Force System

4.1 Centre of Gravity

Since the weight of a body is a system of concurrent forces with the centre of the earth as the point of concurrency, the weight of the body can be represented by the resultant of the system of forces. The point on the body through which this resultant gravitational force passes is known as the centre of gravity of the body. The centre of gravity of a body can be obtained through a very simple experiment. Tie the body at a point by a string and hang it from a support. Then the body will be in equilibrium under the action of two equal, opposite and collinear forces: the weight of the body and the tension in the string. Mark the line of action of the forces by drilling a hypothetical hole of a small diameter. In this way, get two or more holes by tying the body at different points. Interestingly, all the holes would be found passing through a single point, which is the centre of gravity of the body. Therefore, *the centre of gravity of a body can be defined as the point through which the resultant of the elemental gravitational forces of the body would pass regardless of the orientation of the body in the space*. Since the distance between any two elements of a body is usually very very negligible in comparison with the distance from the body to the centre of the earth,

Centre of Gravity

Engineering Mechanics

4.1. Centre of Gravity 47

it may be assumed that the elemental gravitational forces of the body are parallel to each other. Under this assumption, the centre of gravity of a body can be obtained mathematically by applying the principle of moment to a system of parallel forces, which would result the following coordinates for the centre of gravity of the body:

$$\left(\bar{x} = \frac{\int x_c\,dW}{\int dW},\ \bar{y} = \frac{\int y_c\,dW}{\int dW},\ \bar{z} = \frac{\int z_c\,dW}{\int dW}\right) \quad (4.1)$$

Further, since any internal distance of a body is usually negligible in comparison with the distance from the body to the centre of the earth, it may be assumed that the gravitational field on the body is uniform, giving the same value of the gravitational acceleration g at every point of the body, i.e. $dW = d(mg) = gdm$. Therefore, the coordinates of the centre of gravity, given by Eq. (4.1), can be simplified as below:

$$\left(\bar{x} = \frac{\int x_c\,dm}{\int dm},\ \bar{y} = \frac{\int y_c\,dm}{\int dm},\ \bar{z} = \frac{\int z_c\,dm}{\int dm}\right) \quad (4.2)$$

Since the coordinates of Eq. (4.2) contain no reference to the gravitational field g, but only to the mass m of the body, the centre of gravity can be termed as the *centre of mass* under the assumption of a uniform and parallel gravitational field. *Centre of Mass*

Dilip Datta

present case are '**\thechapter. #**1' and '**\thesection. #**1', are internally stored in **\leftmark** and **\rightmark**, respectively (the second argument of **\markboth{}{}** is not processed as mentioned in §5.3.2 on page 42). Various headers and footers, as stated in Table 5.7, are defined in lines 7–11. The arguments of **\lhead[]{}** in line 7 print **\thepage** (page number) and **\rightmark** (information about **\section{}**) as the left aligned headers on even and odd pages, respectively. Similarly, the arguments of **\rhead[]{}** in line 8 print **\leftmark** (information about **\chapter{}**) and **\thepage** (page number) as the right aligned headers on even and odd pages, respectively. Moreover, each of **\thepage**, **\rightmark**, and **\leftmark** is inserted as the argument of **\textbf{}** for printing the corresponding contents in boldface fonts. On the other hand, values to the first argument of **\lfoot[]{}** in line 9 and the second argument of **\rfoot[]{}** in line 10 are supplied for printing their contents as the left aligned footer on even pages and right aligned footer on odd pages, respectively. No value is assigned to the arguments of **\cfoot[]{}** in line 11 in order to keep the center footer blank, otherwise the page number will be printed here also. The **\renewcommand{}{}** command in lines 12 and 13 assigns the value of `0.15mm` to each of **\headrulewidth** and **\footrulewidth** for obtaining rules, each of thickness 0.15 mm, below the header and above the footer, respectively. Moreover, **\addtolength{}{}** is used in lines 14 and 15 for adding the values of **\marginparsep** and **\marginparwidth** to the default value of **\headwidth**, which is done for increasing the horizontal widths of the header, footer and rules to cover the marginal notes also. Finally, **\thispagestyle{empty}** is inserted just after each **\chapter{}** (shown in line 20) to avoid numbering of the first page of a chapter, otherwise **\thispagestyle{plain}** (by default associated with **\chapter{}**) will print the page number as the center footer on the first page of each chapter.

5.3.4 *Header and Footer with the* **fancy** *Style Under the* **fancyhdr** *Package**

Similar to the **fancyheadings** package, there is **fancyhdr** package that also defines the **fancy** page style[1]. The header and footer commands under the **fancyhdr** package are slightly different than those under the **fancyheadings** package. For obtaining the same style (as shown in Table 5.8) under the **fancyhdr** package, the commands of lines 7–11 in Table 5.8 are to be replaced by the six lines of commands given in Table 5.9.

Table 5.9 Header and footer with the **fancy** page style under the **fancyhdr** package

LaTeX input	1	`\fancyhead[le,ro]{\textbf{\thepage}}`
	2	`\fancyhead[re]{\textbf{\leftmark}}`
	3	`\fancyhead[lo]{\textbf{\rightmark}}`
	4	`\fancyfoot[le]{\textbf{Engineering Mechanics}}`
	5	`\fancyfoot[ro]{\textbf{Dilip Datta}}`
	6	`\fancyfoot[c]{}`

[1]The header style under the **headings** page style is predefined, while the header/footer styles under the **myheadings** and **fancy** page styles are user-defined.

The **\fancyhead[ ]{ }** and **\fancyfoot[ ]{ }** are, respectively, the commands for headers and footers, whose **l**, **c**, and **r** options stand for left, center, and right aligned header/footer, while **o** and **e** mean odd and even numbered pages, respectively. Accordingly, the combinations **lo** and **le** stand for left alignment on odd and even numbered pages, respectively, while **ro** and **re** mean right alignment on those pages. If no choice is supplied for alignment, headers/footers will be printed in all the three positions (left, center, and right). Similarly, if no choice is supplied for pages, a header/footer will be printed on both odd and even numbered pages. On the other hand, if none of alignment and page is provided, headers/footers will be printed in all the three alignments on both odd and even numbered pages.

5.4 Page Breaking and Adjustment

The application of the **\nopagebreak** command at a point prevents the page breaking at that point. On the other hand, the **\newpage**, **\pagebreak**, **\clearpage**, or **\cleardoublepage** command may be used for printing the remaining contents of a document on a new page by breaking the current page at the point where the command appears. In the case of **twoside** option to **\documentclass[ ]{ }**, **\cleardoublepage** prints the remaining materials of a document from the next odd numbered (right hand) page, even by leaving the previous even numbered (left hand) page blank, if required. In a multi-column page, the **\newpage** and **\pagebreak** commands start a new column instead of a new page. In that case, a new page can be started using **\clearpage** or **\cleardoublepage** only.

The **\raggedbottom** declaration in the preamble puts texts from the top of a page, leaving extra space at the bottom if no material is available to put in that limited space. The **\raggedbottom** declaration is default, except under the **twoside** option to **\documentclass[ ]{ }**. In contrary, the **\flushbottom** declaration makes all pages of the same height adding extra vertical space, if required[2]. Even when **\flushbottom** is in effect, **\newpage** may be used to produce a shortened page (i.e., like **\raggedbottom** page).

5.5 Page Numbering

As seen in Table 1.2 on page 4, by default LaTeX assigns a page number in Arabic numeral at the bottom-center of every page, starting with 1 from the very first page of a document. Other numbering styles can be obtained by manual setting, either

[2]The **\raggedbottom** declaration (default except **twoside** option to **\documentclass[ ]{ }**) puts texts from the top of a page leaving extra space at the bottom if required, while the **\flushbottom** declaration makes all pages of the same height adding extra vertical space if required.

for all pages or for selective pages of a document. A few such styles are explained below:

1. Five types of page numbering are available, which are listed in Table 5.10[3].

   Table 5.10 Different types of page numbering

Type of numbering	Meaning
arabic	Arabic numerals, like `1`, `2`, …
roman	Lowercase Roman numerals, like `i`, `ii`, …
Roman	Uppercase Roman numerals, like `I`, `II`, …
alph	Lowercase English alphabets, like `a`, `b`, …
Alph	Uppercase English alphabets, like `A`, `B`, …

   A particular type of numbering can be obtained through **\pagenumbering{**`anum`**}**, where `anum` is the required type of numbering as shown in Table 5.10, e.g., **\pagenumbering{alph}** for numbering pages by lowercase English alphabets. The effect would be global if **\pagenumbering{}** is put in the preamble. For local effect, the command can be put inside the **document** environment also, e.g., numbering the pages of the front matter of a book by lowercase Roman numerals, while those of the main matter and back matter by Arabic numerals. If **\pagenumbering{}** is put somewhere inside the **document** environment, the previous numbering style will be altered with effect from the current page onward. Whenever **\pagenumbering{}** is used, the counter of the pages is reset to unity starting from the page containing the command. For effective local implementation, **\pagenumbering{}** should be preceded by one of the **\newpage**, **\clearpage** and **\cleardoublepage** commands, which will terminate the current page and print the remaining materials of a document starting from the next page.

2. By default page numbering starts from unity. Numbering can be started from any other number using **\setcounter{page}{**`n`**}** in the preamble, where `n` (an integer) is the desired starting page number of the document.

3. The **\thispagestyle{empty}** command may be used on a page to avoid it from numbering (although the page will not be numbered, it will be counted while numbering the following pages). This may be useful in a report, where number is not to be shown on the first page. If a page is not to be numbered as well as not to be counted, the **\setcounter{page}{**`n`**}** command with appropriate value of `n` may be used at the starting of the next page.

4. If none of the pages of a document is to be numbered, the **\pagestyle{empty}** or the **\let\thepage\relax** command may be used in the preamble.

[3] Five types of numbering of pages and numbered units are **arabic**, **roman**, **Roman**, **alph**, and **Alph**.

Hour 6

Listing and Tabbing Texts

Important matters in a document are usually listed point-wise, either for concise presentation or for making them prominent. Similarly, texts may also need to be tabbed in different columns along the width of a page.

6.1 Listing Texts

There are three listing environments, namely **enumerate**, **itemize**, and **description**. The **enumerate** environment creates a numbered list and the **itemize** environment creates an unnumbered list, while the **description** environment is used to generate a list with user-defined labels. In any of these environments, each individual item is written through an **\item** command, i.e., an item is preceded by an **\item** command. An **\item** command prints an item on a new line/paragraph, and two items are separated by a predefined vertical gap, which can be controlled locally by assigning a suitable value to the **\itemsep** command inside an environment, e.g., `\setlength{\itemsep}{0mm}` for eliminating the vertical gap, while `\setlength{\itemsep}{10mm}` for maintaining a vertical gap of 10 mm.

6.1.1 Numbered Listing Through the enumerate *Environment*

The **enumerate** environment produces a numbered list of items, where the items are numbered by Arabic numerals as shown in Table 6.1. It is also possible to write

Table 6.1 Numbered listing through the **enumerate** environment

LaTeX input	Output
Some states of India are listed below: \begin{enumerate} \item Assam \item Punjab \item Rajasthan. \end{enumerate}	Some states of India are listed below: 1. Assam 2. Punjab 3. Rajasthan.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_6

an **enumerate** environment inside another **enumerate** environment, in which case the inner environment will belong to an **\item** of the outer environment. A maximum of four **enumerate** environments can be nested one inside another for producing a hierarchy of items[1]. Such an example is shown in Table 6.2, which also shows how

Table 6.2 Nested numbered listing through the **enumerate** environment

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>
Some Asian countries and ... listed below:
\begin{enumerate}
\item India\label{item:Ind}
    \begin{enumerate}
    \item Assam\label{item:Ass}
        \begin{enumerate}
        \item Sonitpur\label{item:Sonit}
            \begin{enumerate}
                \item Tezpur\label{item:Tez}
                \item Dhekiajuli
                \item Balipara
            \end{enumerate}
        \item Kamrup
        \item Cachar
        \end{enumerate}
    \item Bihar
    \item Punjab
    \end{enumerate}
\item Sri Lanka
\end{enumerate}
Place~\ref{item:Tez} ... district~\ref{item:Sonit} in
state~\ref{item:Ass} of country~\ref{item:Ind}.
</pre></td><td>Some Asian countries and their various places are listed below:

1. India
  (a) Assam
    i. Sonitpur
      A. Tezpur
      B. Dhekiajuli
      C. Balipara
    ii. Kamrup
    iii. Cachar
  (b) Bihar
  (c) Punjab
2. Sri Lanka

Place 1(a)iA belongs to district 1(a)i in state 1a of country 1.</td></tr>
</table>

an enumerated item can be labeled and referred through the **\label{}** and **\ref{}** commands, respectively (blank spaces preceding inner lines in the LaTeX input are kept only for easy understanding of a loop, otherwise they do not have any sense in LaTeX). The default numbering styles in the nested four listings under the **enumerate** environment are `1.,2.,...`; `(a),(b),...`; `i.,ii.,...` and `A.,B.,...` respectively, while their referring styles are `1,2,...`; `1a,1b,...`; `1(a)i,1(a)ii,...` and `1(a)iA,1(a)iB,...` respectively.

6.1.1.1 Changing the Numbering Style I*

The default numbering styles in the nested **enumerate** environment can be altered by redefining **\labelenumi**, **\labelenumii**, **\labelenumiii**, and **\labelenumiv**, where **\labelenumi** governs the numbering style in the first **enumerate** environment, **\labelenumii** in the second **enumerate** environment, and so on. Similarly, their default referring styles can be altered by redefining **\theenumi**, **\theenumii**, **\theenumiii**, and **\theenumiv**, respectively. The example of Table 6.2 is reproduced in Table 6.3 on the following page by altering the default numbering and referring styles, where the differences between the two patterns are self-explanatory. The fields **enumi**, **enumii**, **enumiii**, and **enumiv** are, respectively, the counters of the items in four nested

[1] A maximum of four **enumerate** environments can be nested for producing a hierarchy of items.

Table 6.3 Altering styles of numbered listing under the enumerate environment

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td>

```
\documentclass[11pt,a4paper]{article}
%
\renewcommand{\labelenumi}{\arabic{enumi}.}
\renewcommand{\labelenumii}{(\Alph{enumii})}
\renewcommand{\labelenumiii}{(\alph{enumiii})}
\renewcommand{\labelenumiv}{(\roman{enumiv})}
\renewcommand{\theenumi}{\arabic{enumi}}
\renewcommand{\theenumii}{(\Alph{enumii})}
\renewcommand{\theenumiii}{(\alph{enumiii})}
\renewcommand{\theenumiv}{(\roman{enumiv})}
%
\begin{document}
Some Asian countries and ... listed below:
\begin{enumerate}
\item India\label{item:Ind}
    \begin{enumerate}
    \item Assam\label{item:Ass}
        \begin{enumerate}
        \item Sonitpur\label{item:Sonit}
            \begin{enumerate}
                \item Tezpur\label{item:Tez}
                \item Dhekiajuli
                \item Balipara
            \end{enumerate}
        \item Kamrup
        \item Cachar
        \end{enumerate}
    \item Bihar
    \item Punjab
    \end{enumerate}
\item Sri Lanka
\end{enumerate}
Place~\ref{item:Tez} ... district~\ref{item:Sonit} in
state~\ref{item:Ass} of country~\ref{item:Ind}.
\end{document}
```

</td><td>

Some Asian countries and their various places are listed below:

1. India
   (A) Assam
      (a) Sonitpur
         (i) Tezpur
         (ii) Dhekiajuli
         (iii) Balipara
      (b) Kamrup
      (c) Cachar
   (B) Bihar
   (C) Punjab
2. Sri Lanka

Town 1(A)(a)(i) belongs to district 1(A)(a) in state 1(A) of country 1.

</td></tr>
</table>

enumerate environments, while **\arabic{enumi}**., (**\Alph{enumii}**), (**\alph{enumiii}**), and (**\roman{enumiv}**) are the numbering styles of those items.

6.1.1.2 Changing the Numbering Style II*

In many cases some fixed texts may need to be included in the numbering of a set of listed items, e.g., `Exercise`, `Question`, `Note`, etc. Such texts can be included in the redefinitions of the **\labelenumi**, **\labelenumii**, **\labelenumiii** and **\labelenumiv** commands. However, even if the numbering style of the items is changed by including some fixed texts, they will still be referred by their serial numbers only, i.e., without the included fixed texts. Table 6.4 on the next page shows an example where the serial numbers of the items in the first **enumerate** environment are preceded by (`Q.`).

Table 6.4 Numbered listing under the **enumerate** environment mixed with global fixed texts

<table>
<tr><td>LATEX input</td><td>

```
\documentclass[11pt,a4paper]{article}
\renewcommand{\labelenumi}{Q.\theenumi}
%
\begin{document}
The following questions ...\\ Answer Q.\ref{must} and any two from the rest.
\begin{enumerate}
\item Whether the following statements are true or false?\label{must}
    \begin{enumerate}
    \item Water is composed of oxygen and hydrogen.
    \item Scientific symbol of iron is Hg.
    \item The value of the gravitational acceleration is 10.
    \end{enumerate}
\item What is photosynthesis?
\item What do you mean by magnetism?
\item State the Newton's law of motion.
\end{enumerate}
\end{document}
```

</td></tr>
<tr><td>Output</td><td>The following questions are from General Science:
Answer Q.1 and any two from the rest.

Q.1. Whether the following statements are true or false?
(a) Water is composed of oxygen and hydrogen.
(b) Scientific symbol of iron is Hg.
(c) The value of the gravitational acceleration is 10.
Q.2. What is photosynthesis?
Q.3. What do you mean by magnetism?
Q.4. State the Newton's law of motion.</td></tr>
</table>

If the **\labelenumi**, **\labelenumii**, **\labelenumiii** and **\labelenumiv** commands are redefined in the preamble of a document, their effects will be global. To get different local effects in different **enumerate** environments, the commands should be redefined repeatedly before starting every **enumerate** environment. However, it may not always be convenient to redefine the commands every time, particularly when two or more environments are nested one inside another. Such drawbacks can be overcome through the **enumerate** package, which redefines the **enumerate** environment with an optional argument for specifying its numbering style[2], e.g., `\begin{enumerate}[Note 1]` for numbering the items of the environment as `Note 1`, `Note 2`, etc. The tokens **1**, **i**, **I**, **a** and **A** are reserved for indicating a numbering style. If any of these five tokens appears in the fixed texts of the optional field, it is to be protected by writing it in **{}**. For example, the letter '`a`' of '`Lemma`' is to be protected by writing it as `{a}` (i.e., `[Lemm{a}1]`), otherwise it would be treated as a counter instead of just a fixed letter. Although the numbering styles of the items are changed by including some fixed texts, here also the items are referred by their serial numbers only, i.e., without the included fixed texts. Table 6.5 on the following page shows a document that contains multiple **enumerate** environments with different numbering styles.

[2]Numbering styles of items of the **enumerate** environment can be changed, including the addition of some fixed texts, either redefining **\labelenumi**, **\labelenumii**, **\labelenumiii** and **\labelenumiv**, or including an optional argument to the environment as defined in the **enumerate** package.

Table 6.5 Numbered listing under the **enumerate** environment mixed with local fixed texts

LaTeX input	Output

`\documentclass[11pt,a4paper]{article}`
`\usepackage{enumerate}`
`%`
`\begin{document}`
`\begin{center}{\bf EXAMPLES}\end{center}`
`\begin{enumerate}[{\bf Ex{a}mple 1:}]`
`\item Show that...`
`\item Prove that...\label{item:ex_gr}`
`\item What would be...`
`\end{enumerate}`
`%`
`\begin{center}{\bf PROBLEMS}\end{center}`
`\begin{enumerate}[{\bf Problem (a):}]`
`\item Prove that...\label{item:pr_gr}`
`\item Show that...`
`\item What would be...`
`\end{enumerate}`
`%`
`The problem~(\ref{item:pr_gr}) is just an`
`extension of the example~\ref{item:ex_gr}.`
`\end{document}`
	EXAMPLES
	
	Example 1: Show that…
	Example 2: Prove that…
	Example 3: What would be…
	
	PROBLEMS
	
	Problem (a): Prove that…
	Problem (b): Show that…
	Problem (c): What would be…
	
	The problem (a) is just an extension of the example 2.

6.1.2 *Unnumbered Listing Through the* **itemize** *Environment*

An unnumbered list is produced through the **itemize** environment, an example of which is shown in Table 6.6. Unlike in a numbered list, an item of an unnumbered

Table 6.6 Unnumbered listing through the **itemize** environment

LaTeX input	Output

`\begin{itemize}`
`\item Assam`
`\item Bihar`
`\item Punjab`
`\item Rajasthan.`
`\end{itemize}`
	• Assam
	• Bihar
	• Punjab
	• Rajasthan.

list cannot be referred even if it is labeled by a reference key. This is because the item does not have any serial number to refer it. Like the **enumerate** environment, a maximum of four **itemize** environments can be nested one inside another. This is shown in Table 6.7 on the following page, which is the reproduction of the example of Table 6.2 in the **itemize** environments. The default markings of the items in the four nested **itemize** environments are bullet, hyphen, asterisk and dot, respectively.

As in the case of the **enumerate** environment, the default markings in the **itemize** environment can also be altered by redefining the **\labelitemi**, **\labelitemii**, **\labelitemiii** and **\labelitemiv** commands[3]. Table 6.8 on the next page shows an example where the default markings in two nested **itemize** environments are replaced by the symbols ★ and ▷, respectively. As mentioned in §6.1.1.2 on page 51, if **\labelitemi**, **\labelitemii**,

[3]Marking styles of items of the **itemize** environment can be changed by redefining the **\labelitemi**, **\labelitemii**, **\labelitemiii** and **\labelitemiv** commands.

Table 6.7 Nested unnumbered listing through the **itemize** environment

LaTeX input	Output
Some Asian countries and … are listed below: \begin{itemize} \item India \begin{itemize} \item Assam \begin{itemize} \item Sonitpur \begin{itemize} \item Tezpur \item Dhekiajuli \item Balipara \end{itemize} \item Kamrup \item Cachar \end{itemize} \item Bihar \item Punjab \end{itemize} \item Sri Lanka \end{itemize}	Some Asian countries and their various places are listed below: • India – Assam $\ast$ Sonitpur · Tezpur · Dhekiajuli · Balipara $\ast$ Kamrup $\ast$ Cachar – Bihar – Punjab • Sri Lanka

Table 6.8 Altering styles of unnumbered listing under the **itemize** environment

LaTeX input	Output
\renewcommand{\labelitemi}{$\bigstar$} \renewcommand{\labelitemii}{$\Rightarrow$} … \begin{itemize} \item India \begin{itemize} \item Assam \item Bihar \item Punjab \end{itemize} \item Sri Lanka \end{itemize}	★ India ▷ Assam ▷ Bihar ▷ Punjab ★ Sri Lanka

etc., are redefined in the preamble of a document, their effects will be global. On the other hand, if they are redefined somewhere inside the **document** environment, their effects will be local only to those **itemize** environments that follow these redefinitions.

*6.1.3 Listing with User-Defined Labels Through the **description** Environment*

The **description** environment facilitates to prepare a list of items with user-defined labels. Like the **itemize** environment, the **description** environment also does not have any counter, for which its items can not be referred by any serial number. An item in the **description** environment is labeled through an optional argument to the **\item** command, e.g., **\item[**(a)**]** will label its item by **(a)** (labeling of items under the **enumerate** or **itemize** environment can also be changed in this way by providing an optional argument to the **\item** command). Table 6.9 shows an application of the **description**

Table 6.9 Listing with user-defined labels through the **description** environment

LaTeX input	Output
\begin{description} \item[(a)] Assam \item[(b)] Bihar \item[(c)] Punjab \item[(d)] Rajasthan. \end{description}	**(a)** Assam **(b)** Bihar **(c)** Punjab **(d)** Rajasthan.

environment, where its items are labeled by **(a)**, **(b)**, etc. The optional argument of **\item[]** can be anything, like (a), (b), (i), (ii), or Rule, Action, etc., which is printed in boldface fonts. The font style of labeling can be changed by redefining the **\descriptionlabel** command[4], e.g., the **\renewcommand{\descriptionlabel}**[1]**{\textit{#**1**}}** command will print the labels in italic fonts (the effect will be global if redefined in the preamble, otherwise local only to the following **description** environments if redefined inside the **document** environment). Like the **enumerate** and **itemize** environments, the **description** environments can also be nested one inside another.

Since items in the **description** environment are labeled by providing label-names in **[]** after the **\item** command, [] in the starting of an item, if any, is to be protected by writing in **{}** as **{**[]**}**, e.g., '**\item[**Q.1**]** **{**[Delhi/Mumbai]**}** is the capital of India' for producing 'Q.1 [Delhi/Mumbai] is the capital of India'.

6.1.4 Nesting Different Listing Environments

It is discussed in §6.1.1–6.1.3 that two or more of each of the **enumerate**, **itemize** and **description** environments can be nested one inside another. Nesting of different listing environments is also possible for producing a hierarchy of items. Table 6.10 on the next page shows an **enumerate** environment nested separately with another **enumerate**, an **itemize**, and a **description** environments. By default the items of the main **enumerate** environment are numbered by 1, 2 and 3, respectively. Since the first nested environment is another **enumerate** environment, its items are numbered by (a), (b) and (c), respectively, i.e., by the second level of numbering in the nested **enumerate** environments (refer §6.1.1 for detail). However, since different environments are nested in the second case (an **itemize** environment inside an **enumerate** environment), the items of the **itemize** environment are labeled by its first level of labeling, i.e., by bullet marks. On the other hand, as usual the items of the **description**

[4]The font style of item labeling in the **description** environment can be changed by redefining the **\descriptionlabel** command.

Table 6.10 Nested different listing environments

LaTeX input	Output

<pre>\begin{enumerate}
\item SI System
    \begin{enumerate}
    \item Metre
    \item Newton
    \item Second
    \end{enumerate}
\item MKS System
    \begin{itemize}
    \item Metre
    \item Kilogram
    \item Second
    \end{itemize}
\item FPS System
    \begin{description}
    \item[(i)] Foot
    \item[(ii)] Pound
    \item[(iii)] Second
    \end{description}
\end{enumerate}</pre>
	1. SI System
	(a) Metre
	(b) Newton
	(c) Second
	2. MKS System
	• Metre
	• Kilogram
	• Second
	3. FPS System
	(i) Foot
	(ii) Pound
	(iii) Second

environment in the third case are labeled by the supplied texts of `(i)`, `(ii)` and `(iii)`, respectively. These default patterns of labeling can also be altered as discussed in §6.1.1–6.1.3.

6.1.5 *Indentation of Listed Items**

Notice in Tables 6.1, 6.2, 6.3 and 6.7 that the listed items are printed with a predefined indentation on the left side. Sometime space becomes precious seeking to reduce that indentation, which can be done as follows:

▷ The size of item indentation in the **enumerate** and **itemize** environments can be adjusted locally by assigning a suitable value to the **leftmargin** option defined in the **enumitem** package. It is to be done at the starting of an environment, e.g., as `\begin{enumerate}[leftmargin=4mm]` or `\begin{itemize}[leftmargin=4mm]` for adjusting the indentation to 4 mm.

▷ However, the **enumitem** package conflicts with the **enumerate** package, which redefines the **enumerate** environment to take an optional argument for specifying the numbering style of items as explained in §6.1.1.2 on page 51. Accordingly, if both the provisions (changing numbering style and adjusting indentation) are essential in the same document, an alternative for adjusting item indentation in the **enumerate** environment would be to specify the numbering style (not necessarily to change anything) by incorporating `\hspace{}` with a suitable value, e.g., `\begin{enumerate}[\hspace{0mm}1.]` for numbering in Arabic numerals without any indentation, or `\begin{enumerate}[\hspace{0mm}(a)]` for the same job but numbering by lowercase alphabets in a pair of parentheses.

6.2 Tabbing Texts Through the tabbing Environment

The **tabbing** environment is used for aligning texts in different columns. The **\=** command is used, usually in the first row, to generate a new column by ending the current column. The **\>** command moves the control to the next column in the subsequent rows. Each row is terminated by a line break command **** to go to the next row (the last row is not required to be terminated by ****). Table 6.11 shows a simple two-column

Table 6.11 Tabbing texts in different columns through the **tabbing** environment

LaTeX input	Output

`\begin{tabbing}`
`Potato \= 12.00\\`
`Rice \> 20.00\\`
`Oil \> 60.00\\`
`Sugar \> 23.00`
`\end{tabbing}`
	Potato 12.00
	Rice 20.00
	Oil 60.00
	Sugar 23.00

example of tabbing through the **tabbing** environment. The two columns are generated in the first row by a **\=** command (one **\=** command separates two columns) and the row is terminated by ****. The remaining rows are inserted in the same way, but replacing **\=** with **\>**.

6.2.1 Adjusting Column Width in the tabbing Environment

The width of a column is fixed based on the length of the entry in the column in that row in which it is generated. If the width is not sufficient to accommodate the entry of that column in any subsequent row, the **\hspace{}** or **\hspace*{}** command can be used in the column generating row to increase the width of the column. This is shown in Table 6.12, where the width of the first column is increased by 0.5 cm using

Table 6.12 Adjusting tabbing column width in the **tabbing** environment through the **\hspace{}** command

LaTeX input	Output

`\begin{tabbing}`
`Breadth (b)\hspace{0.5cm} \= = 3.00\\`
`Depth (d) \> = 2.00\\`
`Height (h) \> = 4.00\\`
`Volume (V) \> = bdh \= = 24.00\\`
`Base Area (A) \> = bd \> = 6.00\\`
`\end{tabbing}`
	Breadth (b) = 3.00
	Depth (d) = 2.00
	Height (h) = 4.00
	Volume (V) = bdh = 24.00
	Base Area (A) = bd = 6.00

\hspace{0.5cm} in the first row. Without the additional space created by **\hspace{}**, the width of the column would not be sufficient to accommodate the entry of that column in the last row. Note that all the columns in a **tabbing** environment are not required to be generated in the first row itself. In Table 6.12, originally two columns are generated in the first row. The necessity of another (the third) column is felt in the fourth row, and hence it is generated there by splitting the second column into two by using a **\=** command. This column could be generated in the first row also, by adjusting its width through a \hspace{} command. In that case, the third column had to be left blank till the fourth row.

Another option for creating columns of required widths and number is to use the \kill command. In that case, all the columns are generated in the first row itself, where the entry of a column is the widest entry which appears later in that column. Finally, the row is ended by the \kill command, instead of the line breaking command \\, instructing not to print the row but just to generate the columns. As an example of using the \kill command, the list of Table 6.12 is reproduced in Table 6.13.

Table 6.13 Adjusting tabbing column width in the tabbing environment using the \kill command

LaTeX input	Output

\begin{tabbing}
`Base area (A)  \= = 4.00  \= = 24.00\kill`
`Breadth (b)    \> = 3.00\\`
`Depth (d)      \> = 2.00\\`
`Height (h)     \> = 4.00\\`
`Volume (V)     \> = bdh   \> = 24.00\\`
`Base Area (A)  \> = bd    \> = 6.00\\`
\end{tabbing}
	Breadth (b) = 3.00
	Depth (d) = 2.00
	Height (h) = 4.00
	Volume (V) = bdh = 24.00
	Base Area (A) = bd = 6.00

6.2.2 Adjusting Alignment of Columns in the tabbing Environment*

By default the entry of a column in the tabbing environment is left aligned. Provision is there for right-aligning the last entry, for which the last entry is to be preceded by the \` command. Moreover, the \' command can be used between two pieces of texts of the entry of a column, in which case the first piece of texts is printed right aligned in the previous column. The applications of both the \` and \' commands are shown in Table 6.14.

Table 6.14 Aligning tabbing texts in the tabbing environment using \` and \'

LaTeX input	Output
\begin{tabbing} `Longest  \= Longest  \= Longest\kill` `a        \> b        \> c\\` `a        \> b        \> \`c\\` ` \> a\'b \> c\\` `a \> b \> c\\` \end{tabbing}	a b c a b c a b c a b c

Note that \`, \' and \= cannot be used in the tabbing environment for producing accents (refer Table A.1 on page 247 for detail) as they bear different meanings in this environment. Instead of those, \a\`, \a' and \a= are to be used, e.g., the \a\`{o}, \a'{o}, and \a={o} commands in a tabbing environment will generate the accents ò, ó, and ō, respectively, which are the same with those usually produced by the \\`{o}, \'{o} and \={o} commands, respectively.

Hour 7

Table Preparation I

A table is used for presenting data or items row- and column-wise in a concise form. In LaTeX, the **tabular**, **tabularx**, and **longtable** environments are used for preparing different types of tables. However, tables produced by the **tabular** and **tabularx** environments cannot be assigned any serial number or title, which are generally required to identify a table. Moreover, they produce a table as a single object in running texts, which may cause a problem in drawing a big table in the limited space of a page. All such drawbacks can be overcome by nesting the **tabular** and **tabularx** environments with another environment, such as the **table**, **wraptable**, or **sidewaystable** environment (a serial number and a title to the table under the **longtable** environment can be assigned directly).

7.1 Table Through the tabular Environment

Tables are widely prepared through the **tabular** environment, where the columns of a table are generated through the mandatory argument of the environment. For example, **\begin{tabular}{|l|c|c|c|c|}** in Table 7.1 generates a five-column table

Table 7.1 A simple table through the **tabular** environment

LaTeX input

```
\begin{table}[!hbt]
\centering
\caption{Obtained marks.}
\label{tab-marks}
\begin{tabular}{|l|c|c|c|c|}
 \hline Name   & Math & Phy & Chem & English\\
 \hline Robin  & 80   & 68  & 60   & 57\\
 \hline Julie  & 72   & 62  & 66   & 63\\
 \hline Robert & 75   & 70  & 71   & 69\\
 \hline
\end{tabular}
\end{table}
%
Table~\ref{tab-marks} shows the …
```

Output

Table 1: Obtained marks.

Name	Math	Phy	Chem	English
Robin	80	68	60	57
Julie	72	62	66	63
Robert	75	70	71	69

Table 1 shows the marks obtained by three students in the final examination.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_7

(\begin{tabular}[]{} may also be used with optional provision in [] for vertical positioning). A column can be generated through one of the three letters of l, r, and c (other types of columns are discussed in §7.2 and §7.5). Each of these letters represents a column as well as the alignment of the entries in that column (l for left alignment, r for right alignment, and c for center alignment). The | symbol in the argument of \begin{tabular}{} is used either to mark a boundary or to separate two columns by a vertical line in the specified location, covering the full height of the table. Following the \begin{tabular}{} command, the column-wise entries of a row are inserted, separating two entries by an & and ending the row by a line break command \\. Further, the \hline command is used either to mark a boundary or to separate two rows by a horizontal line in the specified location, covering the full width of the table (a \hline command before a row draws a horizontal line above the row). Finally, the tabular environment is ended by the \end{tabular} command (the \hline command just above the \end{tabular} command draws the lower horizontal boundary of the table).

Note that the tabular environment in Table 7.1 is nested inside the table environment for creating the table in a separate paragraph as well as for captioning and labeling it. The table environment is first created through the \begin{table}[!hbt] command (the optional argument !hbt is for the preferred vertical positioning of the table, which is explained in §7.3). The next command in Table 7.1 is \centering, which instructs for width-wise center alignment of the table (other commands could be \flushleft for left alignment or \flushright for right alignment). The \caption{`attl`}[1] command used in the table environment (but outside the tabular environment) assigns a serial number to the table preceded by the default label-word `Table` and followed by a colon, along with its argument `attl` as the title (caption) of the table (since the title usually comes on the top of a table, the \caption{} command is used before the tabular environment). Following the \caption{} command, the \label{} command is inserted with a unique reference key, which as shown in Table 7.1 can be used in the \ref{} command for referring the table anywhere in the document. Also note that \label{} is always used after \caption{}. Moreover, \label{} does not have any effect without \caption{}, in which case the table is not assigned any serial number.

7.2 Table Through the tabularx Environment

In the tabular environment discussed in §7.1, a column is generated by one of the options of l, c, and r. The width of a column under any of these options is made equal to the length of the longest entry in that column. This may extend a table even beyond the width of a page if the table has some very long entries.

The tabularx package provides the tabularx environment, which can calculate automatically the width of a column so as to restrict a table within

[1]If the \caption{`attl`} command is used inside the table, longtable, or sidewaystable environment (but outside the tabular or tabularx environment), the table is assigned a serial number along with the argument `attl` as the title (caption) of the table.

a pre-specified horizontal width irrespective of the lengths of the entries in the table. The **tabularx** environment takes two mandatory arguments, i.e., **\begin{tabularx}{**awidth**}{** acols**}**, where awidth is the horizontal width of the table and acols is its columns. The columns in the **tabularx** environment are generated in the same way as in the **tabular** environment. A fixed-width column is generated through **l**, **c**, or **r**, while a **X** is used to generate a flexible-width column (i.e., a column whose width is to be calculated automatically)[2]. All the flexible-width columns of a table are of equal width, which is calculated internally as the difference of the total width (awidth) of the table and total width of the fixed-width columns, divided by the number of flexible-width columns. Entries in a flexible-width column are made full aligned. Other alignments can be obtained using either **>{\raggedright\arraybackslash}**, **>{\centering\arraybackslash}**, or **>{\raggedleft\arraybackslash}** before **X**, which make the entries left, center, and right aligned, respectively (without the **\arraybackslash** command, the line breaking command **** used for terminating a row may not work properly in some cases). Table 7.2 shows an application of the **tabularx** environment

Table 7.2 A simple table through the **tabularx** environment

LaTeX input

```
\begin{table}[!hbt]
\centering
\caption{Scored points.}
\begin{tabularx}{0.8\linewidth}
                {|X|c|>{\raggedleft\arraybackslash}X|}
 \hline  {\bf Name}  &  {\bf Sex}  &  {\bf Points}\\
 \hline  Milan       &     M       &         1,500\\
         Julie       &     F       &         1,325\\
         Sekhar      &     M       &           922\\
         Dipen       &     M       &           598\\
         Rubi        &     F       &            99\\
 \hline
\end{tabularx}
\end{table}
```

Output

Table 2: Scored points.

Name	**Sex**	**Points**
Milan	M	1,500
Julie	F	1,325
Sekhar	M	922
Dipen	M	598
Rubi	F	99

for generating a three-column table of a total width of 80% of the page width, i.e., 0.8**\linewidth** (a fixed value, say 10cm or 6in, can also be used). Since the middle column is generated by the option **c**, its width is fixed by the longest entry in that column. The extreme two columns are generated by the option **X**, for which their widths are equal and calculated internally to accommodate all the three columns in

[2]In the **tabularx** environment, a fixed-width column is generated through **l**, **c**, or **r**, while a flexible-width column is generated through a **X**.

the pre-specified width (i.e., 0.8\linewidth) of the table. Moreover, the last column is made right aligned by generating it through >{\raggedleft\arraybackslash}X, instead of just through X. All other matters of Table 7.2 are same with those of Table 7.1.

7.3 Vertical Positioning of Tables

As shown in Tables 7.1 and 7.2, the preferred vertical position of a table on a page can be specified as an optional argument to the table environment, i.e., \begin{table}[avp], where avp is the specifier for vertical positioning of the table. The commonly used specifiers are h, b, and t, which stand for here, bottom of the page, and top of the page, respectively. These specifiers can be used individually or in a combination of two or three. Moreover, for placing the table in the specified position even if enough space is not available on the current page, the specifier or the combination of the specifiers may be preceded by a ! symbol, like !h, !b, or !hbt. Irrespective to the order of the specifiers in a combination, LaTeX always uses the following order for positioning a table:

▷ If ! is used, many default or preset restrictions are ignored and a table is attempted to put in the specified position.
▷ If h is given, the table is attempted to put in the exact position. If fails and no more specifier is given, by default LaTeX considers the specifier t for placing the table on the top of the next page.
▷ If t is given, the table is attempted to put on the top of the current page.
▷ If b is given, the table is attempted to put at the bottom of the current page.

Besides h, b, and t, there is another specifier H, which is defined in the float package. Usually, if a table cannot be put on the current page due to space limit, it is taken to the next page and the remaining space of the current page is filled by the texts which are typed in the LaTeX input file after the table. However, H instructs to put a table *here* only. If the blank space on the current page is not sufficient to hold the table, it is taken to the top of the next page along with the texts that follow the table, by leaving the current page incomplete. The specifier H is used alone, i.e., it should not be combined with ! or any of h, b, and t. Refer §8.9 on the page 80 for putting all the tables at the end of a document, regardless of their actual positions in the LaTeX input file.

7.4 Sideways (Rotated) Texts in Tables*

If a table contains some long entries, space can be saved by printing such entries in vertical direction through the sideways environment defined in the rotating package. An application of the sideways environment is shown in Table 7.3 on the next page.

Table 7.3 Table with entries in vertical direction

LaTeX input	Output

`\begin{tabular}{|l|c|c|c|}`
`\hline Name& \begin{sideways}Mathematics\,\end{sideways}&`
`\begin{sideways}Physics\end{sideways}&`
`\begin{sideways}Chemistry\end{sideways}\\`
`\hline Robin & 80 & 68 & 60\\`
`\hline Julie & 72 & 62 & 66\\`
`\hline Robert & 75 & 70 & 71\\`
`\hline`
`\end{tabular}`
	(output table below)

Name	Mathematics	Physics	Chemistry
Robin	80	68	60
Julie	72	62	66
Robert	75	70	71

7.5 Adjusting Column Width in Tables*

The width of a column, generated through **l**, **r**, or **c**, is set automatically based on the length of the longest entry in that column. This may suffer from the drawback of extending a table beyond the page width if the table has some long entries (many users tackle the situation by manually splitting a long entry into multiple rows). On the other hand, the **X** option in the **tabularx** environment generates columns of equal width irrespective of the lengths of their entries. This may also suffer from the disadvantage of allocating excess width to columns having short entries only, while some columns not having sufficient width to accommodate their long entries.

Above drawbacks can be alleviated by generating columns of a table through **p{}**, **m{}**, or **b{}** command defined in the **array** package. The arguments of **p{}**, **m{}**, and **b{}** specify the width of a column, and the letters **p**, **m**, and **b** make an entry, respectively, vertically top aligned, middle aligned, and bottom aligned relative to the alignment in the previous column (entries in all the three cases are horizontally full aligned). Applications of these three options for fixing the width of a column are shown in Table 7.4 on the next page. The same vertical alignment is made to all the three columns in the first three cases. While in the fourth case, three different alignments are made to the three columns of the table. The first column in the fourth case is top aligned (**p{}**), middle one is middle aligned (**m{}**), and the last one is bottom aligned (**b{}**). As a result, the vertical alignment of a column has become relative to that of its previous column. The second column is middle aligned about the top line of the first column which is top aligned. Similarly, the middle of the second column is made the bottom of the third column which is bottom aligned.

Fixing the width of a column by an absolute value, like **p{1.5cm}**, may make a table too small or extending beyond the width of the page (or the column of a multi-column document), particularly if the page or font size is changed in a later stage. Therefore, a good practice would be to fix the width of a column as a fraction of the **\linewidth** command in a single-column document and **\columnwidth** command in a multi-column document[3], e.g., **p{0.3\linewidth}** or **m{0.2\columnwidth}**.

[3] A good practice would be to specify a length as a fraction of **\linewidth** in a single-column document and **\columnwidth** in a multi-column document (instead of a fixed length, like 5**mm**), particularly to avoid any unpleasant output upon changing the page or font size in a later stage.

Table 7.4 Fixing column widths in tables with p{}, m{}, and b{}

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td>\begin{tabular}{|p{1.7cm}|p{1.5cm}|p{1.6cm}|}
\hline This is the first and the ...&
A medium size entry&
This is another long entry\\
\hline
\end{tabular}</td><td><table><tr><td>This is the first and the longest entry</td><td>A medium size entry</td><td>This is another long entry</td></tr></table></td></tr>
<tr><td>\begin{tabular}{|m{1.7cm}|m{1.5cm}|m{1.6cm}|}
\hline This is the first and the ...&
A medium size entry&
This is another long entry\\
\hline
\end{tabular}</td><td><table><tr><td>This is the first and the longest entry</td><td>A medium size entry</td><td>This is another long entry</td></tr></table></td></tr>
<tr><td>\begin{tabular}{|b{1.7cm}|b{1.5cm}|b{1.6cm}|}
\hline This is the first and the ...&
A medium size entry&
This is another long entry\\
\hline
\end{tabular}</td><td><table><tr><td>This is the first and the longest entry</td><td>A medium size entry</td><td>This is another long entry</td></tr></table></td></tr>
<tr><td>\begin{tabular}{|p{1.7cm}|m{1.5cm}|b{1.6cm}|}
\hline This is the first and the ...&
A medium size entry&
This is another long entry\\
\hline
\end{tabular}</td><td><table><tr><td>This is the first and the longest entry</td><td>A medium size entry</td><td>This is another long entry</td></tr></table></td></tr>
</table>

In the columns of a table, entries are printed leaving some blank space on both sides defined by the \tabcolsep command. The length of such a horizontal blank space between two columns can be changed by changing the value of \tabcolsep (default is 6 pt), e.g., \setlength{\tabcolsep}{2mm}. Similarly, the blank space before or after a particular entry can be eliminated using @{}, e.g., \begin{tabular}{|@{}l|l|@{}} will omit blank space on either side of a table, and \begin{tabular}{|l@{}|@{}l|} will omit the blank space between the two columns (@{~} can also be used for leaving a blank space of length equal to that of ~). On the other hand, the indentation of an entry can be increased by redefining the length of the \parindent command (default is 0 pt), e.g., >{\setlength{\parindent}{5mm}}p{} will generate a column, in which entries will be indented by 5 mm.

7.6 Additional Provisions for Customizing Columns of Tables*

Besides the provisions discussed in §7.1–7.5, the tabular and tabularx environments have many more provisions for customizing a table, some of which are outlined here (all of these provisions are defined in the array package).

▷ The style of the entries in a particular column can be altered using >{command} before the column-generating option l, c, r, X, p{}, m{}, or b{}. For example, >{\bfseries}l for printing all the entries of that column in boldface fonts, or >{\centering}p{5cm} for making the entries center aligned.

▷ A column-generating option can be preceded and followed by >{$} and <{$}, respectively, for converting the column into math-mode, e.g., >{$}l<{$} will generate a left-aligned math-mode column so that a mathematical expression can be inserted in that column without creating any more math-mode.

▷ For repeated use of a particular type of column, a new column type can be defined in the preamble through the \newcolumntype{}{} command. For example, \newcolumntype{C}{>{$}c<{$}} can be used for generating directly a center-aligned math-mode column with C, or \newcolumntype{R}{>{\raggedleft\arraybackslash}X} for generating a right-aligned flexible-width column with R.

▷ Instead of repeating a column type for generating multiple number of consecutive columns of the same type, *{n}{ctype} may be used, which means n number of columns of type ctype. For example, \begin{tabular}{|l|*{5}{c|}} will generate a left-aligned column first and then five number of center-aligned columns, with vertical lines on both sides of each column (Table 7.6 on page 67 shows an application).

▷ For changing the width of a column-separating vertical line (default width is 0.4pt), the | sign may be replaced by !{\vrule width aval} with aval as the width of the vertical line, e.g., !{\vrule width 0.9mm} will generate a vertical line of 0.9 mm width.

▷ The width of vertical and horizontal lines created by |, \vline[4], \hline or \cline{} can be controlled by setting the value of the \arrayrulewidth command (default value is 0.4 pt), e.g., \setlength{\arrayrulewidth}{2pt} for obtaining 2 pt thick lines.

▷ The booktabs package provides some commands for drawing horizontal lines of different widths, as well as of different spacings below or above a horizontal line. These commands include \toprule[], \midrule[], \bottomrule[], and \addlinespace[], where the width or spacing value, as applicable, is taken as the argument in []. For example, \toprule[3pt], \midrule[1pt], and \bottomrule[2pt] (instead of \hline) for producing the top, middle, and bottom lines of a table of widths 3 pt, 1 pt, and 2 pt, respectively. On the other hand, \toprule[3pt]\addlinespace[2pt] for leaving 2 pt blank space below the top line, or \addlinespace[1pt]\bottomrule[2pt] for a blank space of 1 pt above the bottom line.

▷ The vertical space between a column entry and a horizontal line, produced by \hline or \cline{} (refer §7.7 for \cline{}), is controlled by the \extrarowheight command defined in the tabularx package. A suitable value can be assigned to \extrarowheight (default value is 0pt) for increasing such space, e.g., \setlength{\extrarowheight}{3mm} for creating an extra space of 3 mm. The \setlength{\extrarowheight}{} command is to be placed before starting the tabular or tabularx environment.

---

[4]The \vline command in the tabular and tabularx environments draws a vertical line, in the place of its application, having a height equal to that of a row.

Table 7.5 Some additional provisions for customizing a table

LaTeX input

```
\begin{table}[!hbt]
\centering
\setlength{\extrarowheight}{4mm}
\begin{tabular}{
            !{\vrule width 0.8mm}>{\bfseries}l|
            >{$}l<{$}!{\vrule width 0.8mm}}
 \hline  Length          &  {\bf In Metre}\\
 \hline  1 Millimeter    &  10^{-3}\\
 \hline  1 Centimeter    &  10^{-2}\\
 \hline  1 Decimeter     &  10^{-1}\\
 \hline  1 Decameter     &  10\\
 \hline  1 Hectometer    &  10^2\\
 \hline  1 Kilometer     &  10^3\\
 \hline
\end{tabular}
\end{table}
```

Output

Length	In Metre
1 Millimeter	10^{-3}
1 Centimeter	10^{-2}
1 Decimeter	10^{-1}
1 Decameter	10
1 Hectometer	10^2
1 Kilometer	10^3

Applications of some of the above provisions in a `tabular` environment are shown in Table 7.5. The `>{\bfseries}` command before the option `l` in the first column prints all the entries of that column in boldface fonts (while `{\bf }` in the second column prints only the heading in boldface fonts). The `>{$}l<{$}` command converts the second column into math-mode, for which the mathematical entries of that column (all other than its heading) could be inserted directly without creating a separate math-mode for an entry. Two vertical lines, each of width 0.8 mm, on both sides of the table have been obtained by the `!{\vrule width 0.8mm}` commands. On the other hand, the `\setlength{\extrarowheight}{4mm}` command before the `tabular` environment creates an extra vertical space of 4 mm above each row of the table.

7.7 Merging Rows and Columns of Tables

When presenting different types of information in a table, some cells are often required to be merged into a single one. The `multirow` package provides the `\multicolumn{}{}{}` and `\multirow{}{}{}` commands for merging two or more columns and rows, respectively. The applications of the commands are shown in Table 7.6 on the facing page.

In `\multicolumn{nc}{calign}{centry}`, n_c is the number of columns to be merged, `calign` is the alignment of the merged column, and `centry` is the entry of that merged cell. Since four columns in the first row in Table 7.6 are merged into a single cell, the number of entries in that row is reduced from six to three (the `\multicolumn{}{}{}` command spanning a single column can also be used for changing the alignment in that column). The permitted `calign` in the `tabular` environment is `l` (for left alignment), `r` (for right alignment), or `c` (for center alignment). Note that the option `X` as `calign` in `\multicolumn{}{}{}` may not work properly under the `tabularx`

Table 7.6 Merging two or more cells of a table into a single one

<table>
<tr><td>LaTeX input</td><td>

```
\begin{tabular}{|l|*{5}{c|}}
\hline \multirow{2}{*}{Name}&  \multicolumn{4}{c|}{Subjects}&
                                    \multirow{2}{*}{Total}\\
 \cline{2-5}        & Math  & Phy  & Chem  & English  &   \\
 \hline
 \hline     Robin   &  80   &  68  &  60   &    57    & 265\\
 \hline     Julie   &  72   &  62  &  66   &    63    & 263\\
 \hline     Robert  &  75   &  70  &  71   &    69    & 285\\
 \hline
\end{tabular}
```

</td></tr>
<tr><td>Output</td><td>
<table>
<tr><td rowspan="2">Name</td><td colspan="4">Subjects</td><td rowspan="2">Total</td></tr>
<tr><td>Math</td><td>Phy</td><td>Chem</td><td>English</td></tr>
<tr><td>Robin</td><td>80</td><td>68</td><td>60</td><td>57</td><td>265</td></tr>
<tr><td>Julie</td><td>72</td><td>62</td><td>66</td><td>63</td><td>263</td></tr>
<tr><td>Robert</td><td>75</td><td>70</td><td>71</td><td>69</td><td>285</td></tr>
</table>
</td></tr>
</table>

environment. Although many people suggest to use `>{\setlength{\hsize}{n_c\hsize}}X` instead of simply `X`, it also may not work properly in some LaTeX compilers. Hence, a good option is to use `p{}` with manually adjusted argument value (the option `X` generates a column using `p{}` internally with automatically adjusted argument value).

Similarly, in `\multirow{n_r}{cwidth}{centry}`, `n_r` is the number of rows to be merged, `cwidth` is the width of the merged cell, and `centry` is the entry of that merged cell. The value of `cwidth` can be set manually (e.g., `25mm` or `1.0in`), or can be obtained an auto-adjusted one using an `*` only. The entry in the merged cell, obtained through `\multirow{}{}{}`, is vertically center aligned. Other alignment can be obtained by assigning an optional argument after the second mandatory argument of the command, e.g., `\multirow{4}{2cm}[3mm]{centry}` for merging four rows to produce a single cell of width 2 cm and to print `centry` in it 3 mm above the vertical center line (a negative value to the optional argument will print `centry` below the vertical center line). When some rows in a column are merged, `\multirow{}{}{}` is used in the first row to be merged and the column in each of the remaining merged rows is left blank (i.e., the column is ended simply by a `&` or `\\`) as shown in the first and last columns in the second row of Table 7.6.

Further, the `\cline{m-n}` command is used in Table 7.6 for drawing a horizontal line covering columns `m` to `n` only. Another noticeable thing in Table 7.6 is the double horizontal line after the heading of the table. This is done using two consecutive `\hline` commands without any line break between them. Similarly, more than one vertical line can also be drawn using additional `|` symbol in the argument of the `\begin{tabular}{}` command.

Note that both the `\multirow{}{}{}` and `\multicolumn{}{}{}` commands can be used together for creating a single cell by merging a number of rows and columns. One such example is `\multicolumn{3}{|c|}{\multirow{2}{*}{Outcome}}`, where three columns and two rows are merged into a single cell for printing ‘`Outcome`’ with center alignment.

7.8 Table Wrapped by Texts*

If the size of a table is very small compared to the width of a page, the **wraptable** environment, supported by the **wrapfig** package, can be used to wrap around the table by texts. The **wraptable** environment needs two mandatory arguments, i.e., `\begin{wraptable}{aside}{asize}`, where `aside` and `asize` are, respectively, the location and size of the table. The location can be specified by **l** (left side of the page) or **r** (right side of the page), while the size is specified in units (e.g., `25mm`, `1.0in`, or `0.3\linewidth`). The **wraptable** environment is similar with the **table** environment; the only difference lies in creating the environment. A self-explanatory application of this environment is shown in Table 7.7.

Table 7.7 Table wrapped by texts through the **wraptable** environment

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>
\begin{wraptable}{r}{5cm}
\centering
\caption{Obtained marks.}
\label{wrap-table}
\begin{tabular}{|l|c|c|c|}
 \hline Name   & Math & Phy & Chem\\
 \hline Robin   & 80 & 68 & 60\\
 \hline Julie   & 72 & 62 & 66\\
 \hline Robert & 75 & 70 & 71\\
 \hline
\end{tabular}
\end{wraptable}
Marks obtained by ... are shown in
Table~\ref{wrap-table}, ...
</pre></td><td>
Table 3: Obtained marks.
<table>
<tr><td>Name</td><td>Math</td><td>Phy</td><td>Chem</td></tr>
<tr><td>Robin</td><td>80</td><td>68</td><td>60</td></tr>
<tr><td>Julie</td><td>72</td><td>62</td><td>66</td></tr>
<tr><td>Robert</td><td>75</td><td>70</td><td>71</td></tr>
</table>
Marks obtained by Robin, Julie and Robert in Math, Phy and Chem are shown in Table 3, which is wrapped around by texts for saving space. However, the total marks obtained by the students could not shown in the table due to space limit. You can find it by hand calculation only.
</td></tr>
</table>

7.9 Table with Colored Background*

In order to make some entries of a table prominent, the **colortbl** package provides the `\rowcolor{}`, `\columncolor{}`, and `\cellcolor{}` commands for coloring, respectively, any row, column, and cell of a table by the color specified as the argument of a command (refer §2.4 on page 13 for detail of colors). For **gray** color, optional provision is there for specifying its intensity also, in which case the commands take the forms of `\rowcolor[gray]{x}`, `\columncolor[gray]{x}`, and `\cellcolor[gray]{x}`, where `x` is the intensity of the **gray** color to be specified by a number between 0 and 1. The row to be colored is to be started with a `\rowcolor{}` command, while a `\cellcolor{}` command is to be entered in the particular cell to be colored. On the other hand, a colored column is to be generated using a `\columncolor{}` command, in the form of `>{\columncolor{}}`, in the argument of the table-generating environment **tabular** or **tabularx**, e.g., `\begin{tabular}{|l|>{\columncolor{green}}c|r|}` command will generate the middle column center aligned and colored by green color.

As a major drawback, \rowcolor{}, \columncolor{}, and \cellcolor{} may override column-separating vertical lines and row-separating horizontal lines in some cases. There is no formal rule to preserve them. Column-separating vertical lines can be preserved by controlling the amounts of color panel overhang on either side of a column. This can be done though two optional arguments to \rowcolor{} and \columncolor{} in the forms of \rowcolor{}[lhang][rhang] and \columncolor{}[lhang][rhang], where lhang and rhang are, respectively, the amounts of overhang on the left and right sides of a column. Without these two optional arguments, a color panel overhangs by default amount of \tabcolsep, while lhang equals rhang if only one is present. On the other hand, for preserving a row-separating horizontal line, \rule{0pt}{rhgt}\noindent may be used, where rhgt is the height of the zero-width rule generated by the \rule{}{} command. Based on some trials, the values of lhang, rhang, and rhgt may be fixed manually, e.g., \rowcolor{green}[0.9\tabcolsep] or \columncolor{blue}[0pt], or \rule{0pt}{2.6ex}\noindent.

Applications of the \rowcolor{}, \columncolor{}, and \cellcolor{} commands, as stated above, are shown in Table 7.8. Because of the repeated application of

Table 7.8 Table with colored background through the \rowcolor{}, \columncolor{}, and \cellcolor{} commands

LaTeX input

```
\begin{table}[!hbt]
\begin{tabular}{|c|l|r|r|}
\hline SN& Item& Price& Amount\\
\hline \rowcolor{red}[0.9\tabcolsep]
      \rule{0pt}{2.9ex}\noindent 1& Rice& 35& 140\\
\hline \rowcolor[gray]{0.7}[0.91\tabcolsep]
      \rule{0pt}{2.7ex}\noindent 2& Sugar& 55& 55\\
\hline \rowcolor{green}[0.91\tabcolsep]
      \rule{0pt}{2.7ex}\noindent 3& Atta& 75& 75\\
\hline \multicolumn{3}{|r|}{Total}&
         \cellcolor{blue} 270\\
\hline
\end{tabular}
\end{table}
```

Output

SN	Item	Price	Amount
1	Rice	35	140
2	Sugar	55	55
3	Atta	75	75
		Total	270

LaTeX input

```
\newcolumntype{B}[2]
      {>{\columncolor{#1}[0.91\tabcolsep]}#2}
\begin{table}[!hbt]
\begin{tabular}{|c|B{gray}{l}|B{red}{r}|B{green}{r}|}
\hline \rowcolor{white} \rule{0pt}{2.8ex}\noindent
         SN& Item& Price& Amount\\
\hline 1& Rice& 35& 140\\
\hline \rowcolor{white} \rule{0pt}{2.8ex}\noindent
         2& Sugar& 55& 55\\
\hline 3& Atta& 75& 75\\
\hline \rowcolor{white} \multicolumn{3}{|r|}{Total}&
         \cellcolor{blue} 270\\
\hline
\end{tabular}
\end{table}
```

Output

SN	Item	Price	Amount
1	Rice	35	140
2	Sugar	55	55
3	Atta	75	75
		Total	270

\columncolor{} in the second example, a new column type with two arguments, B{ccol}{calign}, is defined through \newcolumntype{}[]{}, where ccol is the color argument of \columncolor{} and calign is the alignment of the column to be generated. Note that any of **\rowcolor{}**, **\columncolor{}**, and **\cellcolor{}** overrides their earlier use in a table, which is shown in the second example in Table 7.8, where **\rowcolor{}** overrides **\columncolor{}** and **\cellcolor{}** overrides **\rowcolor{}**.

Also note that the **\rowcolor{}**, **\columncolor{}**, and **\cellcolor{}** commands may not work properly along with the column margin adjusting command **@{}** discussed in §7.5 on page 63 (in such a requirement, however, the **\tabcolsep** command may be redefined).

Hour 8

Table Preparation II

How commonly used tables can be prepared through the **tabular** and **tabularx** environments is discussed in Hour 7 on page 59. Preparation of complicated tables and table-related some high-level issues are discussed in this Hour.

8.1 Nested Tables*

When some materials are to be presented in complicated forms, two or more tables can be nested for entering the materials conveniently, i.e., a separate table can be drawn in a cell of another table. In that case, an inner table will be the entry of a cell of the outer table, and hence it is to be inserted in **{}**. Such an example is shown in Table 8.1 on the following page, where two separate **tabularx** environments are created in two cells (second cells of the second and third rows) of the outer table. In the same way, the **tabular** environments, or a combination of the **tabular** and **tabularx** environments, can also be nested.

8.2 Column Alignment About Decimal Point*

The column-generating options **l**, **c**, **r**, **X**, **p{}**, **m{}**, and **b{}** make all the entries of a column to be aligned either from one side or both sides. However, sometimes the entries of a column may need to be aligned about a particular location of the entries, e.g., numerical data are usually aligned about their decimal marks. The right-aligned option **r** can be used if all the entries contain equal number of decimal digits, like a currency which usually contains two decimal digits. Any other data may contain different numbers of decimal digits, like 50, 2.325, 23.43, etc. In such a situation, the **D{**`asymb`**}{**`aprint`**}{**`adigit`**}** option, defined in the **dcolumn** package, can be used in the **tabular** and **tabularx** environments for generating a column, where `asymb` is the symbol about which the entries are to be aligned, `aprint` is how the symbol is to

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_8

Table 8.1 Nesting two or more tables

LaTeX input

```
\begin{table}[!hbt]
\begin{tabularx}{\linewidth}{|l|X|c|}
\hline Semester& Subject-wise score& Total\\
\hline First& {
        \begin{tabularx}{\linewidth}{X|c}
        English & 69\\
        Science & 80\\
        Drawing & 92
        \end{tabularx}
    }& 241\\
\hline Second& {
        \begin{tabularx}{\linewidth}{X|c}
        English     & 77\\
        Science     & 85\\
        Mathematics & 83\\
        History     & 64
        \end{tabularx}
    }& 309\\
\hline
\multicolumn{2}{|r|}{Grand Total}& 550\\
\hline
\end{tabularx}
\end{table}
```

Output

Semester	Subject-wise score		Total
First	English	69	241
	Science	80	
	Drawing	92	
Second	English	77	309
	Science	85	
	Mathematics	83	
	History	64	
		Grand Total	550

Table 8.2 Aligning columns of a table about decimal marks

LaTeX input

```
\begin{tabular}{|r|D{.}{,}{4}|D{,}{\cdot}{2}|}
 \hline     2354  &    2354  &    2354\\
          25.936  &  25.936  &  25,936\\
          319.48  &  319.48  &  319,48\\
         7.85867  &  7.8586  &  7,8586\\
            4322  &    4322  &    4322\\
 \hline
\end{tabular}
```

Output

2354	2354	2354
25.936	25,936	25·936
319.48	319,48	319·48
7.85867	7,8586	7·8586
4322	4322	4322

be printed in the output, and `adigit` is the maximum number of decimal digits for which space is to be created. Table 8.2 shows three different forms for presenting some data having different numbers of decimal digits. The first column in the **tabular** environment is generated by the option **r**, for which all the data in that column are right aligned irrespective of the number of decimal digits in a data. This is not a good presentation for the obvious reason as seen in the output. The second column is generated by the option **D{.}{,}{4}** to align the data about the '.' mark (first argument), to replace the '.' mark by a ',' (second argument), and to create a space for a maximum four digits after the '.' mark (third argument). Similarly, the third column is generated by the option **D{,}{\cdot}{2}** to align the data about the ',' mark, and to replace the ',' mark by **\cdot**[1], and to create a space for a maximum two digits after the ',' mark. It is seen that, since a space for accommodating a maximum two

[1]The **\cdot** command produces a '·' at the vertical center of a line, unlike a normal period mark that produces a '.' at the bottom level of a line.

digits after the ‘,’ mark was created in the third column, at least the fourth entry which contains four digits after the ‘,’ mark has gone beyond the right margin of the column. Note that if no change is required in the location mark, the same symbol as in the first argument of the **D{}{}{}** option may be used in its second argument also.

8.3 Side-by-Side Tables*

Due to smaller sizes compared with the width of a page, or for the purpose of comparison of data, two or more tables may need to be drawn in a single row, i.e., side-by-side. Table 8.3 shows how individual **tabular** environments can be used for

Table 8.3 Side-by-side tables through consecutive **tabular** environments

LaTeX input

```
\begin{table}[!hbt]
\caption{Marks of 2015 and 2016.}
\begin{tabular}{|l|c|c|}
 \hline Name   & Math & Phy\\
 \hline Robin  &  80  &  68\\
 \hline Julie  &  72  &  62\\
 \hline Robert &  75  &  70\\
 \hline
\end{tabular}\hfill
%
\begin{tabular}{|l|c|c|}
 \hline Name   & Math & Phy\\
 \hline Robin  &  75  &  70\\
 \hline Julie  &  65  &  69\\
 \hline Robert &  78  &  67\\
 \hline
\end{tabular}
\end{table}
```

Output

Table 4: Marks of 2015 and 2016.

Name	Math	Phy
Robin	80	68
Julie	72	62
Robert	75	70

Name	Math	Phy
Robin	75	70
Julie	65	69
Robert	78	67

drawing multiple tables in a single row. Note that there should not be any new line or line break command (e.g., a blank line or a **** command) between two **tabular** environments, otherwise the tables will be drawn one below another. The tables can be separated using the available horizontal space through the **\hfill** command (or manually through the **\hspace{}** command) between each pair of **\end{tabular}** and **\begin{tabular}** commands.

Note that the two side-by-side tables of Table 8.3 are assigned a single serial number and a title as a whole. Such tables produced by consecutive **tabular** environments cannot be assigned individual serial number and title. In order to do that, each table may be prepared in an individual **minipage** environment, so that the tables can be numbered and titled individually by assigning the **\caption{}** command separately to their hosting **minipage** environments[2] (refer §4.4 on page 31 for detail of the **minipage** environment). The side-by-side tables of Table 8.3 are reproduced in Table 8.4, but this time each **tabular** environment is nested inside an individual **minipage** environment, while all the **minipage** environments are nested inside a **table** environment. Further, a **\caption{}** command is used in each **minipage** environment,

Table 8.4 Side-by-side tables through the **minipage** environment

LaTeX input

```
\begin{table}[!hbt]
\begin{minipage}[c]{0.4\linewidth}
\centering
\caption{Marks of 2015.}
\label{1st_table}
\begin{tabular}{|l|c|c|}
 \hline Name    & Math  & Phy\\
 \hline Robin   &  80   &  68\\
 \hline Julie   &  72   &  62\\
 \hline Robert  &  75   &  70\\
 \hline
\end{tabular}
\end{minipage}\hfill
%
\begin{minipage}[c]{0.4\linewidth}
\centering
\caption{Marks of 2016.}
\label{2nd_table}
\begin{tabular}{|l|c|c|}
 \hline Name    & Math  & Phy\\
 \hline Robin   &  75   &  70\\
 \hline Julie   &  65   &  69\\
 \hline Robert  &  78   &  67\\
 \hline
\end{tabular}
\end{minipage}
\end{table}
```

Output

Table 5: Marks of 2015.

Name	Math	Phy
Robin	80	68
Julie	72	62
Robert	75	70

Table 6: Marks of 2016.

Name	Math	Phy
Robin	75	70
Julie	65	69
Robert	78	67

which has assigned individual serial number and title to its table. Moreover, each table can be referred separately using its unique reference key assigned through the **\label{}** command. As between two **tabular** environments in Table 8.3, there should not be any line break or new line command between two **minipage** environments also, otherwise the mini pages will be created one below another.

[2] In order to have individual number and title for side-by-side tables produced in a row, each table may be prepared in an individual **minipage** environment along with assigning a **\caption{}** command separately to each **minipage** environment.

8.4 Sideways (Rotated) Table*

It is discussed in §7.4 on page 62 that some entries in a table can be rotated in the vertical direction through the **sideways** environment defined in the **rotating** package. The same environment can be used for rotating an entire table also (but the caption will remain in the horizontal direction). In this case the table-generating environment, such as **tabular** or **tabularx**, is to be inserted in the **sideways** environment. Such an example is shown in Table 8.5. It would be interesting to see that some entries in the table, which is already rotated vertically through the **sideways** environment, can be rotated horizontally using the same environment. In this case, however, such entries are printed as reflected about the horizontal as shown in Table 8.5.

Table 8.5 Rotated table through the **sideways** environment

LaTeX input

```
\begin{table}[!hbt]
\centering
\caption{Obtained marks.}
\begin{sideways}
\begin{tabular}{|l|*{4}{c|}}
\hline Name& \begin{sideways}Math\end{sideways}&
                    \begin{sideways}Phy\end{sideways}&
                   \begin{sideways}Chem\end{sideways}&
                \begin{sideways}English\end{sideways}\\
 \hline Robin   & 80 & 68 & 60 & 57\\
 \hline Julie   & 72 & 62 & 66 & 63\\
 \hline Robert  & 75 & 70 & 71 & 69\\
 \hline
\end{tabular}
\end{sideways}
\end{table}
%
Table~\ref{tab-marks} shows the marks ...
```

Output

Table 7: Obtained marks.

Name	Math	Phy	Chem	English
Robin	80	68	60	57
Julie	72	62	66	63
Robert	75	70	71	69

Table 7 shows the marks obtained by students.

The table shown in Table 8.5 through the **sideways** environment is produced on the same portrait-size page along with other texts of the document. However, a big table, which cannot be accommodated along the width of a portrait page, may need to be drawn on a landscape-size page. Such tables are drawn through the **sidewaystable** environment defined in the **rotating** package. Unlike in the **table** environment, the preferences for the vertical location of a table do not work in the **sidewaystable** environment. This is because the **sidewaystable** environment draws a table on a new page in landscape-mode and no other text is permitted to be printed on that page. An example of the **sidewaystable** environment is shown in Table 8.6 on the next page (output is not shown).

Table 8.6 Rotated table on landscape page through the **sidewaystable** environment

```
\begin{sidewaystable}
\centering
\caption{Marks obtained by three students in different subjects.}
\begin{tabular}{|*{7}{c|}}
 \hline  Name   & Physics & Mathematics & Chemistry & Biology & English & History\\
 \hline  Robin  &   80    &     68      &    60     &   75    &   70    &   71\\
 \hline  Julie  &   72    &     62      &    66     &   80    &   68    &   60\\
 \hline  Robert &   75    &     70      &    71     &   72    &   62    &   66\\
 \hline
\end{tabular}
\end{sidewaystable}
```

8.5 Long Table on Multiple Pages*

Sometimes a lot of information may need to be presented in a single table, which cannot be accommodated on a single page or in the remaining space of the current page. In that case, the **longtable** environment, defined in the **longtable** package, may be used for preparing a table which is expandable through multiple pages[3]. An example of a long table stretching over two pages is shown in Table 8.7 on the next page, whose various points are explained below:

1. A separate **tabular** environment is not required for generating the table. It is done in the **longtable** environment itself.
2. The columns of the table are defined as the mandatory argument of the **\begin{longtable}[ ]{ }** command, whose optional argument takes one of the letters **l**, **c**, and **r** as the alignment of the table (**l** for left aligned, **c** for centered, and **r** for right aligned). By default a **longtable** is center aligned.
3. The **\caption{ }** command is followed by the line break command ****.
4. The header of the table is created in two steps: header for the first page and header for the successive pages. The header of the first page is ended by **\endfirsthead** and that of the successive pages is ended by **\endhead**. The header of the first page contains the **\caption{ }** command and the headings of the columns. Similarly, the header of the successive pages also contains the continued caption of the table and the headings of the columns. Since the **\caption{ }** command cannot be repeated in a single table, the caption in Table 8.7 for the successive pages is generated manually through the syntax `\multicolumn{5}{c}{\tablename\ \thetable: First Year Marks (contd\ldots)}`. The `\multicolumn{5}{c}{ }` command is used for producing a centered caption covering all five columns of the table. The commands **\tablename** and **\thetable** print, respectively, the label-word and serial number of the table.

[3] A table prepared in the **longtable** environment, defined in the **longtable** package, is split automatically over multiple pages if it cannot be accommodated on a single page or in the remaining space of the current page.

Table 8.7 Long table in multiple pages through the **longtable** environment

LaTeX input

```latex
Unlike other years, … subjects are given in Table~\ref{marksheet}.
%
\begin{longtable}[c]{|l|*{4}{r|}}
\caption{First Year Marks.}\\
\hline  Name& Math& Phy& Chem& Total\\
\hline  \endfirsthead
\multicolumn{5}{c}{\tablename\ \thetable:
                First Year Marks (contd\ldots)}\\
\hline  Name& Math& Phys& Chem& Total\\
\hline  \endhead
\hline  \multicolumn{5}{r}{Continued on the next page\ldots}\\
\endfoot
\hline  \endlastfoot
\label{marksheet}
%       End of headers and footers
 Akash Singh    & 70  & 75  & 60  & 205\\
 Badal Batra    & 82  & 79  & 66  & 227\\
 .
 .
 .
 Uttam Bhadawal & 81  & 76  & 73  & 230\\
 Vikram Seth    & 83  & 85  & 78  & 246\\
\end{longtable}
%
It is sure that not only the students, …
```

Output

Unlike other years, the students of the first year students have shown the best result in this year. The total marks of all the students in Mathematics, Physics and Chemistry have crossed 200, and in percentage it is more than 68%. Since Mathematics, Physics and Chemistry are the main subjects considered for engineering course, if they can continue such performance in the second year also, certainly a good number of students will be able to join an engineering course. Their current marks in these three subjects are given in Table 1.

Table 1: First Year Marks.

Name	Math	Phy	Chem	Total
Akash Singh	70	75	60	205
Badal Batra	82	79	66	227
Chandan Choudhury	89	82	85	256
Deepak Oja	69	72	76	217
Eswar Prasad	71	78	69	218
Fatema Begum	76	80	71	227
Gautom Vir Singh	92	80	78	250
Harish Negi	81	70	68	219
Indra Sharma	77	71	66	214
Jeevan Lal	80	75	71	226
Kavita Sen	79	71	72	222
Lakshmi Devi	93	82	79	254
Madhav Saikia	88	74	65	227
Nagesh Dahal	72	70	71	213
Onkar Sridhar	87	80	82	249
Prakash Bhatta	77	72	71	220

Continued on the next page…

1

Table 1: First Year Marks (contd…)

Name	Math	Phy	Chem	Total
Queen Hazarika	83	76	72	231
Radhika Misra	82	78	75	235
Sekhar Surmawala	76	68	67	211
Tanveer Azmi	84	80	75	239
Uttam Bhadawal	81	76	73	230
Vikram Seth	83	85	78	246

It is sure that not only the students, but the teachers will also be happy th such performance of their students of a single batch. It would be a pride r the Institute as well.

2

5. Following the headers are the footers of the table for printing some information at the bottom of the table. Usually on each page, other than on the last page, a footer is put to show that the table is continued on the following page (it is done in Table 8.7 by `\multicolumn{5}{r}{Continued on the next page\ldots}`). The footer of a page (other than the last page) is ended by the **\endfoot** command, while that of the last page is ended by the **\endlastfoot** command.
6. The table can be labeled through the **\label{}** command after the headers and footers of the table.

If a table appears on the middle of a page and the remaining space of the page is not sufficient to hold it, a normal table will be printed on the next page. Therefore, a table may be prepared in the **longtable** environment if it is suspected to be big enough (having many rows) to come on a single page or in a part of a page as shown in Table 8.7. If the table comes on a single page, it will be printed just like other normal tables. In that case, the **\endhead** and **\endfoot** commands will become inactive.

8.6 Tables in Multi-column Documents

In a multi-column document, where texts are printed in multiple columns on a page (refer §4.3 on page 29 for multi-column documents), a table is also placed in a column. However, if the width of the column is not large enough to accommodate a table in it, the **table*** environment may be used for drawing the table over the entire width of the page[4]. In that case, the **\begin{table}** and **\end{table}** commands are to be replaced by the **\begin{table*}** and **\end{table*}** commands, respectively.

8.7 Foot Notes in Tables*

In the **tabularx** and **longtable** environments, a foot note against a word/phrase can be generated by inserting the **\footnote{}** command after it with the foot note as the argument of the command. However, the **\footnote{}** command cannot be used directly inside the **tabular** environment. A foot note in the **tabular** environment can be generated through the **\footnotemark** and **\footnotetext{}** commands in the same way as in the case of a mini page (refer Table 4.7 on page 35). However, in this way only one foot note can be generated per **tabular** environment. If more than one foot notes are to be generated, the **tabular** environment may be put inside a **minipage** environment and then the **\footnote{}** command can be used to generate foot notes as many required. In this case, the foot notes will be printed just below the mini page. Examples of both the cases for generating foot notes in the **tabular** environment are shown in Table 8.8 on the facing page, where the foot note in the first **tabular** environment is prepared through the **\footnotemark** and **\footnotetext{}** commands, and hence it is printed at the bottom of the main page. On the other hand, the two foot notes in the second **tabular** environment inside a **minipage** environment are prepared through the **\footnote{}** command, and hence these are printed just after the mini page was over (refer §4.5.1 on page 33 for more detail).

[4]If the column width of a multi-column document is not sufficient to accommodate a table in it, the **table*** environment may be used for drawing the table over the entire width of the page.

Table 8.8 Foot notes in a table under the **tabular** environment

LaTeX input

```
\begin{tabular}{|l|l|}
 \hline Sonitpur\footnotemark & Tezpur\\
        Kamrup                & Guwahati\\
        Dibrugarh             & Dibrugarh\\
 \hline
\end{tabular}
\footnotetext{Sonitpur is in Assam}
%
\begin{minipage}[t]{5cm}
\begin{tabular}{|l|l|}
\hline
 Assam\footnote{Assam is in East}       & Dispur\\
 Tripura\footnote{It is also in East} & Agartala\\
 Nagaland                               & Kohima\\
\hline
\end{tabular}
\end{minipage}
```

Output

Sonitpur[1]	Tezpur
Kamrup	Guwahati
Dibrugarh	Dibrugarh

Assam[a]	Dispur
Tripura[b]	Agartala
Nagaland	Kohima

[a] Assam is in East
[b] It is also in East

[1] Sonitpur is in Assam

8.8 Changing Printing Format of Tables*

The default printing format of a table, as shown in Table 7.1 on page 59, can be changed in different ways. Four such changes are discussed here:

- ▷ If the **\caption{}** command is used in the **table** environment, LaTeX assigns a serial number to the table, which is preceded by the default label-word '`Table`'. It can be replaced by any other word(s) by inserting the **\def\tablename{}** command in the preamble with the desired word(s) as the argument of the command, e.g., **\def\tablename{**`Tab.`**}** will replace `Table` by '`Tab.`'.
- ▷ The type and size of fonts for the label-word and caption can also be changed using the **\captionsetup{}** command defined in the **caption** package (the **\captionsetup{}** command is also to be inserted in the preamble). As an example, **\captionsetup{margin=**`10`**pt, font=it, labelfont={large, bf, sf}}** may be used for printing the caption in 10 point italic fonts, and the label-word in large and boldface Sans serif fonts (because of more than one in number, the values of **labelfont** are inserted in **{}**).
- ▷ By default a single-line caption is center aligned, while a multi-line caption is full justified. In order to fully justify even a single-line caption, the **justification=justified** and **singlelinecheck=false** options may also be included in **\captionsetup{}**.
- ▷ If excess vertical blank space is left before or after the caption of a table, the **\abovecaptionskip** and **\belowcaptionskip** commands may be inserted in the preamble for skipping such blank spaces.

A noticeable thing in this book is that the serial number of a table (e.g., Table 8.1 on page 72) is composed of two parts, `8` and `1` separated by a period, where `8` is the serial number of the chapter (Hour in this book) and `1` is the serial number of the table in that chapter. In contrast, a table in the document-class **article** is assigned its serial number only, i.e., not preceded by the serial number of the section in which the table belongs (§11.4.3 on page 105 and §19.2.5 on page 189 discuss the process for obtaining section-wise serial numbers to tables in the document-class **article**).

8.9 Tables at the End of a Document

Some publishers want the tables and figures of an article to be grouped at the end of the article. Just the inclusion (loading) of the **endfloat** package in a normal document automatically performs this job, regardless of the actual positions of the tables and figures in the LaTeX input file[5] (the **endfloat** package produces two auxiliary files with `fff` and `ttt` extensions for writing information about the figures and tables, respectively). Not only the tables and figures are grouped at the end of the document, but also the notes are produced in their actual positions, like `[Table 3 about here.]` or `[Figure 7 about here.]`. Moreover, the tables and figures are preceded by two lists, namely '`List of Tables`' and '`List of Figures`', respectively, containing their contents. Note that the use of the **endfloat** package may require an additional `latex` run to move the tables and figures at the end of the document (§15.4 on page 149 and §16.2.3 on page 160 discuss about the `latex` run).

The produced lists of tables and figures can be turned off using the **\notablist** and **\nofiglist** commands in the preamble, in which case the tables and figures will be placed just under the headings of '`Tables`' and '`Figures`', respectively. Similarly, the notes in the actual positions of tables and figures can be turned off by putting the **\nomarkersintext** command in the preamble. On the other hand, the formats of the notes can be changed by redefining the **\tableplace** and **\figureplace** commands, e.g., the following two redefinitions will produce notes, like `[Table 3 is at the end of the article]` and `[Figure 7 is at the end of the article]`, respectively:

```
\renewcommand{\tableplace}%
   {[\tablename~\theposttbl\ is at the end of the article]}
```

and

```
\renewcommand{\figureplace}%
   {[\figurename~\thepostfig\ is at the end of the article]}
```

[5] Just the inclusion (loading) of the **endfloat** package automatically puts the tables and figures at the end of a document, regardless of their actual positions in the LaTeX input file.

Hour 9

Figure Insertion

LaTeX has the provision for inserting a figure from an external file in different formats. As stated in §1.4 on page 4, a LaTeX file can be compiled using either the latex or pdflatex command. When a LaTeX file involves figures from external files, either of the compilation commands is to be used based on the format of the figures. Note that the file formats of all the figures inserted in a LaTeX document must be supported by a single compilation command, either latex or pdflatex. Commands for compiling LaTeX files involving some standard and widely used figure formats are given in Table 9.1. It is to be mentioned that different tools, like `xfig` and `gimp` on Unix system or `ImageMagick` and `netpbm` on both Unix and Windows systems, can be used for exporting figures from one format to another.

Table 9.1 LaTeX compilation commands and supported figure formats

Compilation command	Supported figure format: Short name	Full name
latex	`eps`	Encapsulated PostScript
	`ps`	PostScript
pdflatex	`pdf`	Portable Document Format
	`jpeg`	Joint Photographic Expert Group
	`tiff`	Tag Index File Format
	`png`	Portable Network Graphic

9.1 Commands and Environment for Inserting Figures

An `eps` format figure can be inserted using the **\epsfig{file=`fname`}** command defined in the **epsfig** package, where `fname` is the name of the figure file with or without the '`eps`' extension. Apart from the mandatory `fname`, the size of a figure can also be specified in **\epsfig{}** through two optional fields, **width** and **height**, one separated from another by a comma. Without any of the **width** and **height**, a figure is printed in its original size. If one of them is specified, the other one

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_9

is automatically taken in proportion. On the other hand, the presence of both **width** and **height** prints a figure in the specified fixed size (in this case, the figure may get distorted if their values are not set properly). In addition to specifying the size, a figure can also be rotated through the option **angle=**`theta`, where a positive value of `theta` (in degree) will rotate the figure in counter-clockwise direction and a negative value in clockwise direction. With such provisions, a figure can be inserted as **\epsfig{file=**`myfig.eps`**}** or **\epsfig{file=**`pics/myfig.eps`**, width=**`0.5`**\linewidth}** or **\epsfig{file=**`myfig.eps`**, width=**`30`**mm, height=**`40`**mm, angle=**`30`**}**, where `myfig.eps` is the name of the figure file and `pics` is the folder containing the figure file.

The more general command for inserting a figure from an external file is **\includegraphics[**`aopt`**]{**`fname`**}** defined in the **graphicx** package, where `fname` is the name of the figure file without its extension, and `aopt` is(are) the option(s) like **width**, **height** and **angle**. The advantage of using **\includegraphics[]{}** is that a figure in any format can be inserted without making any change in the LaTeX input file[1].

Similar to nesting the **tabular** or **tabularx** environment in the **table** environment as discussed in §7.1 on page 59, the **\epsfig{}** and **\includegraphics[]{}** commands can be used in the **figure** environment, so that a figure can be assigned a serial number and a caption through the **\caption{}** command, as well as a reference key through the **\label{}** command for the purpose of referring it anywhere within a document. Further, similar to the **table** environment, the **figure** environment can also be created as **\begin{figure}[]** with optional preferences in **[]** for vertical positioning of a figure. The standard preferences for vertical positioning are **H**, and any or combination of **h**, **b** and **t** along with **!** (refer §7.3 on page 62 for detail of **[H]** and **[!hbt]**).

9.2 Inserting a Simple Figure

Three examples of inserting a figure, named as `girl.eps`, through the **\epsfig{}** command are shown in Table 9.2 on the next page. In the first example, the size of the figure is specified by both width and height, which have produced the figure in a distorted form due to the consideration of their non-proportionate values. In the second example, since only the width of the figure is specified, a scaled form of the original figure is produced by automatically adjusting its height. In the third example, apart from specifying the width of the figure, it is also rotated by 30^{o} in the counter-clockwise direction.

The first command in the **figure** environment in Table 9.2 is **\centering**, which instructs for width-wise center alignment of its figure (other commands could be **\flushleft** for left alignment or **\flushright** for right alignment). After inserting the figure through **\epsfig{}**, the **\caption{}** command is used for assigning a serial number to the figure. The **\caption{}** command will also produce a title of the figure, if any is provided as the argument of the command (since the title usually comes at the bottom of a figure, **\caption{}** is used after **\epsfig{}**). The **\caption{}** command is followed

---

[1] The advantage of using the **\includegraphics[]{}** command for inserting figures from external files is that a figure in any format can be inserted without making any change in the LaTeX input file.

Table 9.2 Figure insertion through the **\epsfig{}** command

LaTeX input	Output
\begin{figure}[!hbt] **\centering** **\epsfig{file=**`girl.eps`**, width=**`2.0`**cm, height=**`2.0`**cm}** **\caption{**`A girl.`**}** **\label{**`girl1`**}** **\end{figure}**	 Figure 1: A girl.
\begin{figure}[!hbt] **\centering** **\epsfig{file=**`girl.eps`**, width=**`2.0`**cm}** **\caption{**`A girl.`**}** **\label{**`girl2`**}** **\end{figure}**	 Figure 2: A girl.
\begin{figure}[!hbt] **\centering** **\epsfig{file=**`girl.eps`**, width=**`2.0`**cm, angle=**`30`**}** **\caption{**`A girl.`**}** **\label{**`girl3`**}** **\end{figure}**	 Figure 3: A girl.

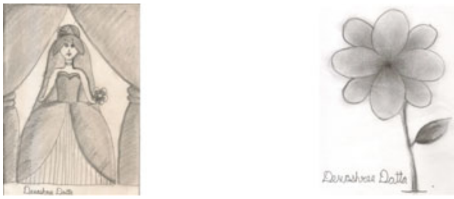
by the **\label{}** command for assigning a unique reference key, which can be used for referring the figure through the **\ref{}** command. Note that **\label{}** does not have any effect without **\caption{}**, in which case a figure will not be assigned any serial number for referring it.

The same outputs, shown in Table 9.2, can be produced through the **\includegraphics[]{}** command also. In that case, just the **\epsfig{}** command from the examples is to be replaced by **\includegraphics[width=**`2`**cm, height=**`2`**cm]{**`girl`**}**, **\includegraphics[width=**`2`**cm]{**`girl`**}** and **\includegraphics[width=**`2`**cm, angle=**`30`**]{**`girl`**}**, respectively. Since a figure in any format can be inserted through **\includegraphics[]{}** without making any change in the LaTeX input file, now onwards all figures in this book will be inserted through **\includegraphics[]{}** only, otherwise mentioned specifically.

9.3 Side-by-Side Figures*

In the examples in §9.2, only one figure is inserted in a row. Provision is also there in LaTeX for inserting multiple figures side by side in a single row. As

Table 9.3 Side-by-side figures in a single row

LaTeX input	Output
\begin{figure}[!hbt] \centering \includegraphics[width=2.0cm]{girl}\hfill \includegraphics[width=2.0cm]{flower} \caption{A girl and a flower.} \label{girl_flower} \end{figure}	 Figure 4: A girl and a flower.

shown in Table 9.3, this can simply be done by inserting each figure through a separate **\includegraphics[ ]{ }** in a single **figure** environment. The requirements for the same are: there should not be any line break or new line command between two **\includegraphics{ }**, and total width of all the figures should not exceed the page width; otherwise the figures will be inserted one below another. However, **\hfill** can be used between two **\includegraphics{ }** for separating the corresponding figures by available horizontal space.

Note that both the side-by-side figures in Table 9.3 are assigned a single serial number as a whole. If the side-by-side figures are to be assigned individual serial number, the **minipage** environment may be used. As stated in §8.3 on page 73, the **minipage** environment splits a page width-wise into a number of parts, each of which can be used for inserting a figure, drawing a table, or even for writing selected texts. Table 9.4 shows two figures, inserted side-by-side by using the **minipage** environment, which are assigned individual serial number by each **\caption{ }** command. Within a single **figure** environment, two **minipage** environments, each of size 0.4**\linewidth**, are created for inserting the two figures.

Table 9.4 Side-by-side figures through the **minipage** environment

LaTeX input	Output
\begin{figure}[!hbt] \begin{minipage}[c]{0.4\linewidth} \centering \includegraphics[width=3.0cm]{girl} \caption{A girl.} \label{girl} \end{minipage}\hfill % \begin{minipage}[c]{0.4\linewidth} \centering \includegraphics[width=2.5cm]{flower} \caption{A flower.} \label{flower} \end{minipage} \end{figure}	 Figure 5: A girl.  Figure 6: A flower.

9.4 Sub-numbering a Group of Figures

In some cases, a group of figures may need to be sub-numbered under a main number, e.g., 3(a) or 5(e). Within the **figure** environment, the **\subfigure[**atitle**]{**afig**}** command defined in the **subfigure** package[2] (or the new **\subfloat[**atitle**]{**afig**}** command defined in the **subfig** package) may be used for inserting a figure with a sub-numbering, where optional `atitle` is the title of the figure, and mandatory `afig` is the insertion of the figure either through the **\epsfig{}** or **\includegraphics[]{}** command. For the purpose of referring, a subfigure can be assigned a reference key through the **\label{}** command inside the mandatory argument of **\subfigure[]{}**. Moreover, the group of subfigures can be captioned and labeled as a whole using respectively the **\caption{}** and **\label{}** commands inside the **figure** environment. Table 9.5

Table 9.5 Sub-numbering a group of figures using the **\subfigure[]{}** command

LaTeX input	Output
\begin{figure}[!htb] \centering \subfigure[A girl.] { \includegraphics[width=2.0cm]{girl} \label{girl} }\hfill \subfigure[A flower.] { \includegraphics[width=2.0cm]{flower} \label{flower} }\\ \subfigure[A finger work.] { \includegraphics[width=4.0cm]{finger} \label{finger-work} } \caption{Girl, flower and finger work.} \label{girl_flower_finger} \end{figure} % In Figure~\ref{girl_flower_finger}, \ref{girl} and \ref{flower} display a girl and a flower, while \ref{finger} displays a beautiful finger work.	  (a) A girl. (b) A flower.  (c) A finger work Figure 7: Girl, flower and finger work. In Figure 7, 7(a) and 7(b) display a girl and a flower, while 7(c) displays a beautiful finger work.

shows such an example, which contains three figures in a group. It is also shown in Table 9.5 that the subfigures can be inserted in a single row or even in multiple rows (for inserting a subfigure in the next row, a line break command **** is to be used at the end of the previous **\subfigure[]{}** command).

---

[2]The **subfigure** package loaded as **\usepackage[tight]{subfigure}** with the **tight** option reduces the excess vertical blank space between a subfigure and its caption (the function of this **tight** is similar with those of the **\abovecaptionskip** and **\belowcaptionskip** commands addressed in §8.8 and §9.10).

The \subfigure[]{} command numbers a group of subfigures as (a), (b), etc. This default numbering can be changed by redefining the \thesubfigure command, e.g., \renewcommand{\thesubfigure}{(\roman{subfigure})~} for numbering subfigures by (i), (ii), etc. Note that a subfigure is not assigned any sub-number without the optional argument to \subfigure{}. However, it is counted while numbering the remaining subfigures of the group.

9.5 Figure Wrapped by Texts*

Like a table as stated in §7.8 on page 68, a figure can also be wrapped around by texts, which is done through the wrapfigure environment[3] defined in the wrapfig package. The wrapfigure environment, which creates a space for inserting a figure, takes one optional and two mandatory arguments, i.e., \begin{wrapfigure}[aline]{aside}{asize}, where aside is the location of the space (l for left side or r for right side), while asize is the size of horizontal space and aline is the number of lines of texts for vertical space. If insufficient number of lines (i.e., aline) is opted, the figure will be overlapped with other lines of texts. If number of lines to be wrapped is not opted, it is set automatically to cover the size of the figure. An example of the wrapfigure environment is shown in Table 9.6, where

Table 9.6 Figure wrapped by texts

LaTeX input	Output

`\begin{wrapfigure}[10]{r}{2.3cm}`
`\includegraphics[width=2.0cm]{girl}`
`\caption{Girl.}`
`\label{girl}`
`\end{wrapfigure}`
`%`
`The picture shown on the right side was drawn by Devoshree, a 8 year old girl....`
	The picture shown on the right side was drawn by Devoshree, a 8 year old girl. She loves arts from her childhood, starting from the age of around 2 years. Seeing her sketching something on a wall and enlarging it every day at that age, we did not stop her but encouraged for the same. Today she can do very beautiful sketching and other artistic works. Only three simple works from her activities are included in this Hour.
	
	Figure 8: Girl.

a figure of width 2.0 cm is inserted through \includegraphics[]{} in a horizontal space of 2.3 cm on the right side of the page wrapping 10 lines of texts of the document.

[3]The wrapfigure environment may not work properly inside other environments. Moreover, the environment should be put at the top of a paragraph, or between two words where a line break exists.

9.6 Rotated Figure

It is shown in Table 9.2 on page 83 that a figure can be rotated by a given angle through the **angle** option to the **\epsfig{}** (also to **\includegraphics[]{}**) command. Further, like the **sidewaystable** environment for producing a table on a landscape-size page as discussed in §8.4 on page 75, the **sidewaysfigure** environment can be used for producing a figure in landscape-mode on a new page. An example of the **sidewaysfigure** environment is shown in Table 9.7 (output is not shown).

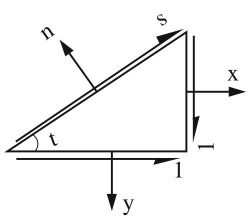
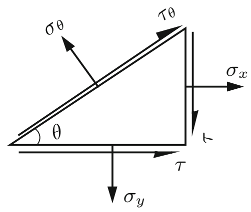
Table 9.7 Rotated figure on a landscape page through the **sidewaysfigure** environment

```
\begin{sidewaysfigure}
\includegraphics[width=\linewidth]{computer}
\caption{A computer on a landscape page.}
\end{sidewaysfigure}
```

9.7 Mathematical Notations in Figures*

There is always a question how to put mathematical notations in figures, like σ_x or e^z. Generally graphics plotters do not have such provisions, but for plain texts and symbols only. The **\psfrag{**atag**}{**acomm**}** command, defined in the **psfrag** package, can replace an alphabetical or a numerical tag (atag) in an encapsulated postscript (eps format) figure with an arbitrary instruction including LaTeX commands (acomm). An application of **\psfrag{}{}** is shown in Table 9.8, in which the alphabetical and

Table 9.8 Mathematical notations in figures through the **\psfrag{}{}** command

Original figure	LaTeX input	LaTeX output
	\begin{figure}[!hbt] \centering \psfrag{x}{σ_x} \psfrag{y}{σ_y} \psfrag{n}{σ_{θ}} \psfrag{s}{τ_{θ}} \psfrag{t}{θ} \psfrag{1}{τ} \includegraphics[width=5cm]{psfrag} \end{figure}	

numerical tags of the original eps figure shown in the left column are replaced, in the figure shown on the right column, by the LaTeX commands inserted as the second argument of the **\psfrag{}{}** commands shown in the middle column. Note that the **\psfrag{}{}** commands are to be inserted before inserting the figure.

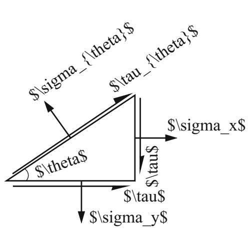
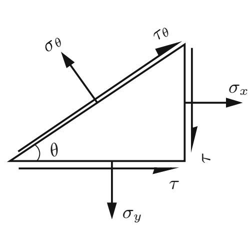
If a figure is prepared in the xfig software, LaTeX commands can directly be inserted in the .fig file. Then the required .eps file can be obtained from the .fig file using the fig2eps command in a terminal, e.g., '$ fig2eps mypic.fig' for generating `mypic.eps` from `mypic.fig`. However, the process needs the `fig2ps` software installed in the computer.

Alternatively, if the `fig2ps` software package is not available, the `fig` file may be exported in a different way to generate a `.pstex_t` file, the contents of which can then be inserted (pasted) in the LaTeX input file in order to achieve the desired effect. The step-by-step procedure of this process is given below:

1. Draw the figure in the `xfig` software package.
2. Click the text-mode '`T`' button on the top-left side of the `xfig` window. Then click the '`Text Flags hidden=off`' button on the bottom side of the `xfig` window, which will open a small dialogue box. In that dialogue box, change the status of '`Special Flag`' from '`Normal`' to '`Special`' and then click '`Set`'.
3. Now typeset LaTeX commands in the required locations of the figure. Math-mode commands are to be inserted between a pair of **$** symbols.
4. Save the file by a name, say `myfig.fig`.
5. Click '`Export`' under the '`File`' button in the menu bar on the top of the `xfig` window, which will open a new dialogue box. Under '`Language`' on the top side of that dialogue box, select '`Combined PS/LaTeX (both parts)`' and then click '`Export`' button.
6. Two new files now should be available in the working folder, `myfig.pstex` and `myfig.pstex_t` (if the figure file in Step (5) above was named by `myfig.fig`).
7. Open `myfig.pstex_t` in a text-editor, and then copy and paste its contents in the required location of the LaTeX input file.

An application of inserting mathematical notations in figures by putting LaTeX commands in the `xfig` software is shown in Table 9.9, in which the LaTeX commands of

Table 9.9 Mathematical notations in figures through `xfig` software

Original .fig figure	**Contents of .pstex_t file**	**LaTeX output**
	**\begin{picture}(**0,0**)** **\includegraphics{**myfig.pstex**}%** **\end{picture}%** **\setlength{\unitlength}{**3947sp**}%** ⋮ ⋮ **\end{picture}%**	

the original `fig` figure shown in the left column are executed, in the figure shown on the right column, by inserting the contents of the `myfig.pstex_t` file in the LaTeX input file as shown in the middle column. Since directly copied and pasted, it is not necessary to understand the contents of the self-generated `.pstex_t` file.

9.8 Figures in Tables*

Sometimes a figure may need to be produced in a cell of a table. In that case also, the figure can be inserted in the cell through **\includegraphics[ ]{ }** just like an ordinary entry in a cell of a table. However, since the **figure** environment cannot be nested inside the **table** environment, **\includegraphics[ ]{ }** cannot be put in a **figure** environment, and hence, the figure can neither be captioned nor numbered. A number of examples can be found in this book, where figures are inserted in tables, refer, e.g., Tables 9.2, 9.3, 9.4, 9.5 and 9.6.

9.9 Figures in Multi-column Documents

In a multi-column document, where the contents are produced in multiple columns on a page, a figure is also placed in a column (refer §4.3 on page 29 for detail of multi-column documents). However, if the width of the column is not large enough to accommodate a figure in it, the **figure*** environment can be used for inserting the figure on the entire width of the page[4]. In that case, the **\begin{figure}** and **\end{figure}** commands are to be replaced by the **\begin{figure*}** and **\end{figure*}** commands, respectively.

9.10 Changing Printing Format of Figures*

The serial number of a figure is preceded by the default label-word '`Figure`', which can be changed by defining **\def\figurename{}** in the preamble, e.g., **\def\figurename{**`Fig.`**}** will replace '`Figure`' by '`Fig.`'. Moreover, the size and type of fonts for the label-word and caption can also be changed by using **\captionsetup{}** as discussed in §8.8 on page 79. Similarly, **\abovecaptionskip** and **\belowcaptionskip**, mentioned in §8.8 for reducing the excess vertical blank space before and after the main caption of a table, is applicable in the case of figures also.

In the document-class **book**, the serial number of a figure is composed of two parts. For example, refer Fig. 4.1 on page 27, where the figure number is composed of 4 and 1 separated by a period mark. In this case, 4 is the number of the Hour and digit 1 is the number of the figure in that Hour. In contrast, a figure in the document-class **article** is assigned its serial number only, i.e., not preceded the number of the section in which the figure belongs (§11.4.3 on page 105 and §19.2.5 on page 189 discuss the process for obtaining section-wise numbers to figures in the document-class **article**).

[4]If the column width of a multi-column document is not sufficient to accommodate a figure in it, the **figure*** environment may be used for inserting the figure over the entire width of the page.

9.11 Figures at the End of a Document

Refer §8.9 on page 80 for detail.

9.12 Editing LaTeX Input File Involving Many Figures*

The compilation time of a LaTeX file involving a lot of figures may be very large. Hence, the **\psdraft** command may be inserted in the preamble when the file needs repeated editing and compilation. Instead of producing a figure, the command instructs the LaTeX compiler just print the name of the figure file in a box in the specified location, thus saves compilation time. However, **\psdraft** is to be deleted or commented before the final compilation. In an alternate way, the **epsfig** or **graphicx** package, as applicable, may be loaded with the **draft** option as **\usepackage[draft]{epsfig}** and **\usepackage[draft]{graphicx}** (or jointly as **\usepackage[draft]{epsfig,graphicx}**) for performing the same job. In this case also, the **draft** option is to be omitted before the final compilation.

Figure Drawing

How a figure can be inserted in a LaTeX document is discussed in Hour 9 on page 81. Besides importing a figure from an external file, LaTeX provides many commands and environments for directly drawing different types of geometric figures.

A geometric figure may be a single or a combination of various smaller elements, like lines and curves. For drawing a figure, a space is first divided into four quadrants by two rectangular coordinate axes (x- and y-axes). The intersection of the axes is called the origin, whose coordinate is (0,0). As shown in Fig. 10.1, x value is positive on the right side of the origin and negative on its left side, while y value is positive above the origin and negative below it. Before drawing a figure, note the following five points:

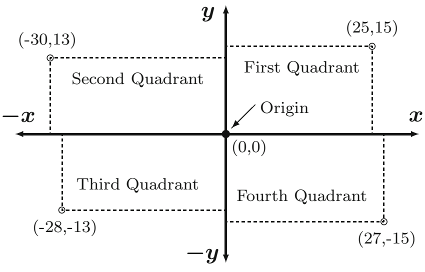


Fig. 10.1 Coordinate axes for drawing figures

1. For drawing figures, a rectangular widow is first reserved through the **picture** environment as **\begin{picture}(**`lx,ly`**)(**`x0,y0`**)**, where (x_0, y_0) is the lower left coordinate of the window, and l_x and l_y are its lengths along the x- and y-axes, respectively (unlike usual LaTeX commands, which takes an argument in **{}** or **[ ]**, figure-related commands and environments take coordinates and lengths in **( )**, two values separated by a comma). For example, **\begin{picture}(**`50,40`**)(**`0,0`**)** reserves a window for drawing figures with $x \in [0, 50]$ and $y \in [0, 40]$, or **\begin{picture}(**`45,35`**)(**`-20,15`**)** for drawing figures with $x \in [-20, (45 - 20)]$ and $y \in [15, (35 + 15)]$.
2. The scale of drawing can be set by defining the **\unitlength** command outside the **picture** environment, e.g., **\setlength{\unitlength}{**`5`**mm}** for setting 1 unit = 5 mm (millimeter). Other acceptable units are **cm** (centimeter), **in** (inch), **pt** (printer point, 1 in = 72.27 pt, 1 cm = 28.45 pt), **em** (width of `M`), and **pc** (pica, 1 pc = 12 pt).

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_10

3. The thickness of a line can be controlled by the **\linethickness{}** command, e.g., **\linethickness{**0.5**mm}** for lines of 0.5 mm thickness. Alternatively, the **\thinlines** and **\thicklines** commands may be used directly for thin and thick lines, respectively. These commands can be used in the **picture** environment, and can also be repeated for drawing lines of different thicknesses.
4. Many figure drawing commands do not ask for the starting coordinate of a figure, and by default the figure is started from the current coordinate. Such a command, say fcmd, can be forced by the **\put(**x,y**){**fcmd**}** command to start the figure at (x,y). The **\multiput(**x,y**)(**Δx,Δy**){**n**}{**fcmd**}** command can also be used for drawing the same figure n times, starting the first one at (x,y) and incrementing (x,y) each time by (Δx,Δy) for the subsequent figures (**\multiput()(){}{}** is a very convenient command for drawing equidistant parallel lines)[1].
5. The **picture** environment does not support the **\caption{}** command. Hence, the **picture** environment may be put in the **figure** environment with the **\caption{}** and **\label{}** commands, as shown in Hour 9, for the purpose of assigning a serial number to the figure drawn and referring it in the document.

10.1 Circles and Circular Arcs

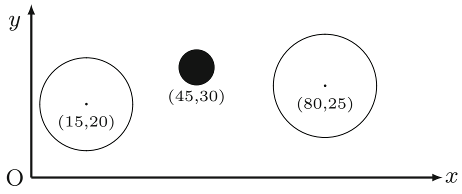
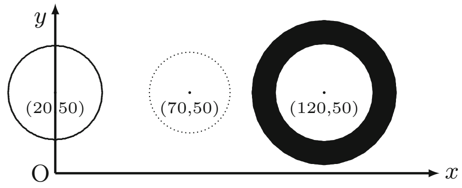
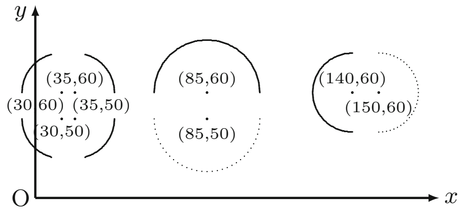
The most easiest figure in LaTeX is a circle, which is drawn by the **\circle{**d**}** or **\circle*{**d**}** command (both are defined in the **pict2e** package), where d is the diameter of the circle. The **\circle{}** command draws a hollow circle, while the **\circle*{}** command makes it solid. Another available command for drawing a circle is **\bigcircle[**p**]{**d**}** defined in the **curves** package, where d is the diameter and optional p is any nonnegative integer representing the pattern (type of line) of the circle (the default value of p is 0). The **\bigcircle[]{}** and **\circle{}** commands can also be used for drawing half-filled circles by increasing the line thickness through **\linethickness{}**.

On the other hand, the **\arc[**p**](**x', y'**){**deg**}** command, defined in the **curves** package, draws a circular arc starting at $(x' + x_c, y' + y_c)$ and moving anticlockwise through the angle deg given in degree (negative value for deg can be used for clockwise movement), where (x_c, y_c) is the coordinate of the center of the arc. That is, (x', y') is a relative coordinate that assumes (0, 0) as its center coordinate (x_c, y_c). If not provided (through **\put(){}**), the current coordinate is taken as (x_c, y_c). As in **\bigcircle[**p**]{}**, the optional p in **\arc[**p**](){}** represents the pattern of the arc.

Since none of **\circle{}**, **\circle*{}**, **\bigcircle[]{}** and **\arc[](){}** asks for the coordinate of the center of a circle or an arc, the figure may be drawn through **\put(**x_c, y_c**){**acomm**}**, where acomm is the figure drawing command and (x_c, y_c) its center coordinate. Some examples of these four commands are shown in Table 10.1 on the following page, where the effects of **\linethickness{}** and **\thinlines** on the line thicknesses are also demonstrated. In Table 10.1, the center points of the

[1]The **\multiput(**x,y**)(**Δx,Δy**){**n**}{}** command can be used for drawing the same figure n times, say equidistant parallel lines, starting the first one at (x,y) and incrementing (x,y) each time by (Δx,Δy) for the subsequent figures.

Table 10.1 Circle and circular arc drawing

LaTeX input	Output
\setlength{\unitlength}{0.5mm} \begin{picture}(120,45)(-10,0) \put(15,20){\circle{25}} \put(45,30){\circle*{10}} \linethickness{2mm} \put(80,25){\circle{30}} \end{picture}	
\begin{picture}(150,70)(20,15) \thinlines \put(20,50){\bigcircle{35}} \put(70,50){\bigcircle[4]{30}} \linethickness{3mm} \put(120,50){\bigcircle{45}} \end{picture}	
\begin{picture}(170,70)(20,20) \thinlines \put(35,60){\arc(15,0){75}} \put(30,60){\arc(-15,0){-75}} \put(35,50){\arc(15,0){-75}} \put(30,50){\arc(-15,0){75}} % \put(85,60){\arc(20,0){180}} \put(85,55){\arc[4](20,0){-180}} % \put(140,60){\arc(0,15){180}} \put(150,60){\arc[4](0,-15){180}} \end{picture}	

circles and arcs are encircled by small circles as well as their coordinates are shown for illustrative purpose only (the commands for the same are not shown in the LaTeX input file). Note that figures can be drawn in different colors also, e.g., \put(15,25){\textcolor{red}{\circle{20}}} will draw a circle in red color.

10.2 Straight Lines and Vectors*

A straight line is drawn by the \line(l_x, l_y){f} command, where l_x and l_y are the least nondivisible factors of the end coordinate (x_e, y_e) of the line (i.e., slope of the line), and f is their greatest common factor (i.e., horizontal or vertical length of the line, horizontal for $l_x \neq 0$ and vertical for $l_x = 0$). When multiplied by f, (l_x, l_y) gives the end coordinate, i.e., (l_xf, l_yf) = (x_e, y_e). The application of \line(){} is slightly complicated. The terms l_x and l_y must obey the following three rules:

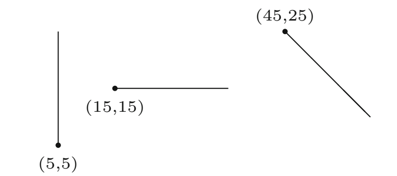
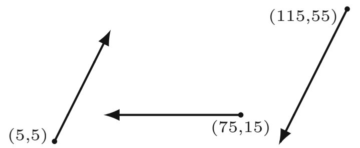
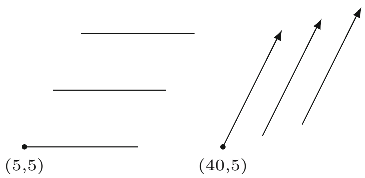
- ▷ Both l_x and l_y are whole numbers, including 0, either positive or negative.
- ▷ Only limited combinations of 0, ±1, …, ±6 in (−6,6) are permitted to l_x and l_y.

▷ l_x and l_y should not have any common factor, like (2,4) or (6,-2) are not permitted. However, (±1,±1) are permitted. In general, the possible combinations of (l_x,l_y) are (1,0), (1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (2,3), (2,5), (3,4), (3,5), (4,5), and (5,6), including their altered combinations and negative counterparts.

Drawing a horizontal or a vertical line is easy. For a horizontal line, the command is simplified to \line(±1,0){hlen}, where hlen is the true length of the line, toward right for \line(1,0){} and toward left for \line(-1,0){}. Similarly, the command for a vertical line is \line(0,±1){hlen}, where the line moves upward for \line(0,1){} and downward for \line(0,−1){}. While seeking horizontal and vertical lines, even if non-unity values are assigned to l_x and l_y (like \line(3,0){} or \line(0,−5){}), they will be treated internally as $l_x = \pm 1$ and $l_y = \pm 1$.

Similar to \line(l_x,l_y){f} for drawing a straight line, the command for drawing a vector (a straight line with an arrow at one end) is \vector(l_x,l_y){f} with l_x,l_y $\in(-4, 4)$. Like the commands of Table 10.1 used for drawing circles or circular arcs, \line(){} and \vector(){} also do not ask for the starting coordinate of a line or a vector. Therefore, their starting coordinates may be specified through \put(){} or \multiput()(){}{}. Some examples of \line(){} and \vector(){} are shown in Table 10.2,

Table 10.2 Straight line and vector (arrow) drawing

LaTeX input	Output
\setlength{\unitlength}{0.75mm} \begin{picture}(60,30)(0,0) \put(5,5){\line(0,1){20}} \put(15,15){\line(1,0){20}} \put(45,25){\line(1,-1){15}} \end{picture}	
\begin{picture}(120,60)(0,0) \thicklines \put(5,5){\vector(1,2){20}} \put(75,15){\vector(-1,0){50}} \put(115,55){\vector(-1,-2){25}} \end{picture}	
\setlength{\unitlength}{0.75mm} \begin{picture}(60,30)(0,0) \multiput(5,5)(5,10){3}{\line(1,0){20}} \multiput(40,5)(7,2){3}{\vector(1,2){10}} \end{picture}	

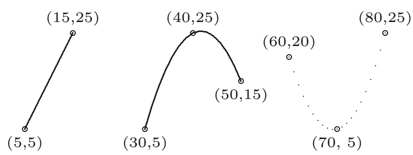
where the starting points of the lines and vectors are encircled by small circles as well as their coordinates are shown for illustrative purpose only.

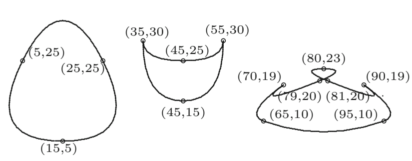
10.3 Curves*

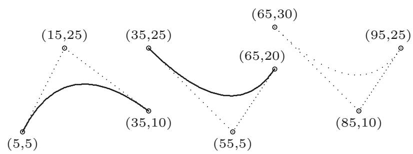
Frequently used commands for drawing curves are **\curve[p](**x_1,y_1, ..., x_n,y_n**)** and **\closecurve[p](**x_1,y_1, ..., x_n,y_n**)**, defined in the **curves** package, which draw respectively open and closed curves through the given coordinate points with the optional `p` as the patterns of the curves. In the **\curve[ ]( )** command, two points draw a straight line and three points draw a parabola. The **\closecurve[ ]( )** command, where at least three points are required, draws a closed curve with continuous tangents at all the points. Besides these two commands, there is **\qbezier[N](**x_1,y_1**)(**x_2,y_2**)(**x_3,y_3**)** for drawing a quadratic Bézier curve through the end points (x_1,y_1) and (x_3,y_3), where the middle point (x_2,y_2) is known as the control point and it is the intersection of the two tangents to the curve at the two end points. The optional argument `N` instructs to approximate the curve through (`N+1`) points. If `N` is not provided, its value is calculated automatically to produce a solid curve. Some examples of **\curve[ ]( )**, **\closecurve[ ]( )** and **\qbezier[ ]( )( )( )** are shown in Table 10.3, where the curve

Table 10.3 Open and closed curves, and Bézier quadratic curves

LaTeX input	Output

`\setlength{\unitlength}{0.7mm}`
`\begin{picture}(80,30)(4,0)`
`\curve(5,5, 15,25)`
`\curve(30,5, 40,25, 50,15)`
`\curve[10](60,20, 70,5, 80,25)`
`\end{picture}`
	

`\setlength{\unitlength}{0.7mm}`
`\begin{picture}(110,40)(4,0)`
`\closecurve(5,25, 15,5, 25,25)`
`\closecurve(35,30, 45,15, 55,30, 45,25)`
`\closecurve(65,10, 70,19, 79,20, 80,23,`
`                81,20, 90,19, 95,10)`
`\end{picture}`
	

`\setlength{\unitlength}{0.7mm}`
`\begin{picture}(100,35)(0,0)`
`\qbezier(5,5)(15,25)(35,10)`
`\qbezier(35,25)(55,5)(65,20)`
`\qbezier[20](65,30)(85,10)(95,25)`
`\end{picture}`
	

generating points are encircled by small circles as well as their coordinates are shown for illustrative purpose only (these are not shown in the LaTeX input file). Since all the relevant coordinates are provided to **\curve[ ]( )**, **\closecurve[ ]( )** and **\qbezier[ ]( )( )( )**, the **\put( ){ }** command, as used in Tables 10.1 and 10.2, is not required here to specify any more point.

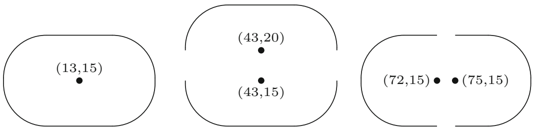
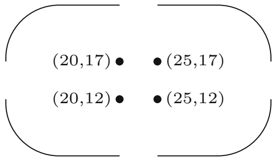
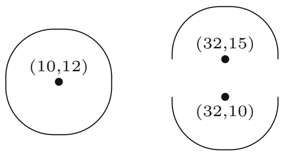
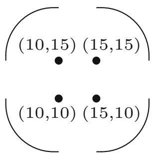
10.4 Oval Boxes*

The **\oval(**$\mathtt{l_x}$**,** $\mathtt{l_y}$**)[**`part`**]** command draws a box of dimensions $\mathtt{l_x}$ and $\mathtt{l_y}$ with rounded corners. The `part` option allows to draw only a part of a box, whose permissible values are given in Table 10.4.

Table 10.4 Options to the **\oval( )[ ]** command for drawing partial ovals

Option	Meaning
t	Top-half of the box.
b	Bottom-half of the box.
l	Left-half of the box.
r	Right-half of the box.
tl or **lt**	Top-left quarter of the box.
tr or **rt**	Top-right quarter of the box.
bl or **lb**	Bottom-left quarter of the box.
br or **rb**	Bottom-right quarter of the box.

Table 10.5 Oval boxes (boxes with rounded corners) through the **\oval( )[ ]** command

LaTeX input	Output
\setlength{\unitlength}{1mm} \begin{picture}(88,30)(0,0) \put(13,15){\oval(25,15)} \put(43,20){\oval(25,15)[t]} \put(43,15){\oval(25,15)[b]} \put(72,15){\oval(25,15)[l]} \put(75,15){\oval(25,15)[r]} \end{picture}	
\setlength{\unitlength}{1mm} \begin{picture}(45,30)(0,0) \put(20,17){\oval(30,15)[tl]} \put(25,17){\oval(30,15)[tr]} \put(20,12){\oval(30,15)[bl]} \put(25,12){\oval(30,15)[br]} \end{picture}	
\setlength{\unitlength}{1mm} \begin{picture}(40,25)(0,0) \put(10,12){\oval(14,14)} \put(32,15){\oval(14,14)[t]} \put(32,10){\oval(14,14)[b]} \end{picture}	
\setlength{\unitlength}{1mm} \begin{picture}(25,25)(0,0) \put(10,15){\oval(14,14)[tl]} \put(15,15){\oval(14,14)[tr]} \put(10,10){\oval(14,14)[bl]} \put(15,10){\oval(14,14)[br]} \end{picture}	

Some applications of **\oval( )[ ]** are shown in Table 10.5, where a box is put in a specified base point through the **\put( ){ }** command. As before, the base points by small solid circles along with their coordinates are also shown for illustrative purpose. Since no option for any part (i.e., the optional `part`) is provided, the very first **\put(){\oval()}** in each of the first and third examples draws a complete box with the base point as its center coordinate. Any other command in Table 10.5 draws a part of a box due to the presence of one of the options as given in Table 10.4. Note that, whether a complete box or a part of it is opted, the complete size of the box (l_x, l_y) is to be provided to **\oval()[]**, which draws the box or its part with the base point, specified through **\put(){}**, as the center coordinate of the complete box. It is observed that for equal values of l_x and l_y, in some small amount (a maximum of around 14 mm), **\oval()[]** draws a circle-like box or a part of it. Some such examples are also shown in the last two examples of Table 10.5.

10.5 Texts in Figures*

If required, some texts can be inserted in a figure through **\put(**x,y**){**atext**}** in the **picture** environment, where `atext` is the texts to be inserted and (x,y) is its lower left coordinate. Additionally, there are three box making commands, **\makebox(**l_x,l_y**)[**pos**]{**atext**}**, **\framebox(**l_x,l_y**)[**pos**]{**atext**}** and **\dashbox{**dsize**}(**l_x,l_y**)[**pos**]{**atext**}**, for conveniently inserting texts in a rectangular box positioned by **\put(){}**. The fields l_x and l_y are respectively the horizontal and vertical lengths of a box, `atext` is the texts to be inserted in the figure, and `dsize` is the size of the dashes in a dashed box. The optional argument `pos` is the vertical and horizontal positions of `atext` in the box, whose permissible values are given in Table 10.6.

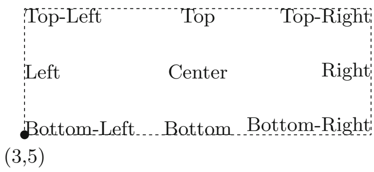
Table 10.6 Settings of the **\makebox()[]{}**, **\framebox()[]{}** and **\dashbox{}()[]{}** commands for positioning texts in figures

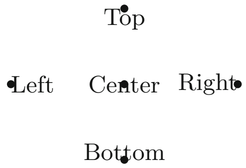
Argument	Meaning
t	Vertically top aligned and horizontally centered.
b	Vertically bottom aligned and horizontally centered.
c	Centered both vertically and horizontally (default value).
l	Horizontally left aligned and vertically centered.
r	Horizontally right aligned and vertically centered.
s	Horizontally aligned on both the edges and vertically centered.
tl or **lt**	Top-left corner of the box.
tr or **rt**	Top-right corner of the box.
bl or **lb**	Bottom-left corner of the box.
br or **rb**	Bottom-right corner of the box.

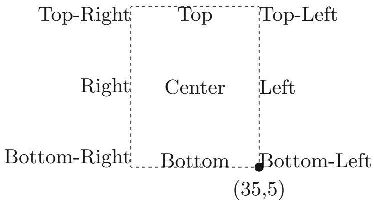
No visible box is produced by **\makebox()[]{}**, it only reserves a rectangular area for printing texts. In the first example in Table 10.7 on the next page, a box of size 55 mm×20 mm is created a number of times at point (3, 5) by **\put(){\makebox()[]{}}** and variations in the positions of texts with the `pos` option are shown. The base coordinate (3, 5) and a dotted box are also shown for illustrative purpose. The **\makebox()[]{}** command has two special properties. First, a zero-dimension box

Table 10.7 Texts in figures through the `\makebox( )[ ]{ }` command

LaTeX input	Output

`\setlength{\unitlength}{1mm}`
`\begin{picture}(60,30)(0,0)`
`\put(3,5){\makebox(55,20)[c]{Center}}`
`\put(3,5){\makebox(55,20)[t]{Top}}`
`\put(3,5){\makebox(55,20)[b]{Bottom}}`
`\put(3,5){\makebox(55,20)[l]{Left}}`
`\put(3,5){\makebox(55,20)[r]{Right}}`
`\put(3,5){\makebox(55,20)[tl]{Top-Left}}`
`\put(3,5){\makebox(55,20)[tr]{Top-Right}}`
`\put(3,5){\makebox(55,20)[bl]{Bottom-Left}}`
`\put(3,5){\makebox(55,20)[br]{Bottom-Right}}`
`\end{picture}`
	

`\setlength{\unitlength}{1mm}`
`\begin{picture}(50,30)(0,0)`
`\put(25,25){\makebox(0,0)[t]{Top}}`
`\put(25,15){\makebox(0,0)[c]{Center}}`
`\put(25,5){\makebox(0,0)[b]{Bottom}}`
`\put(10,15){\makebox(0,0)[l]{Left}}`
`\put(40,15){\makebox(0,0)[r]{Right}}`
`\end{picture}`
	

`\setlength{\unitlength}{1mm}`
`\begin{picture}(50,30)(0,0)`
`\put(35,5){\makebox(-20,25)[c]{Center}}`
`\put(35,5){\makebox(-20,25)[t]{Top}}`
`\put(35,5){\makebox(-20,25)[b]{Bottom}}`
`\put(35,5){\makebox(-20,25)[l]{Left}}`
`\put(35,5){\makebox(-20,25)[r]{Right}}`
`\put(35,5){\makebox(-20,25)[tl]{Top-Left}}`
`\put(35,5){\makebox(-20,25)[tr]{Top-Right}}`
`\put[35,5){\makebox(-20,25)[bl]{Bottom-Left}}`
`\put(35,5){\makebox(-20,25)[br]{Bottom-Right}}`
`\end{picture}`
	

can also be created, in which case a piece of texts is positioned with respect to the base coordinate of `\makebox( )[ ]{ }`, some examples of which are shown in the second part of Table 10.7. Second, a box of a negative dimension can also be created, which means the negative side of the base point. As shown in the third part of Table 10.7, a box of size (−20, 25 mm) means 20 mm left and 25 mm above the base point. Such a negative-dimensional box may be useful if a piece of texts is to be inserted on the left or below the base point.

Unlike `\makebox( )[ ]{ }`, `\framebox( )[ ]{ }` produces a visible box of a specified size. Moreover, no negative or zero-dimensional box is permitted in `\framebox( )[ ]{ }`. If no size is provided, a box of an arbitrary size is created. All other issues of `\framebox( )[ ]{ }` are the same with those of `\makebox( )[ ]{ }`. The `\dashbox{ }( )[ ]{ }` command is similar with `\framebox( )[ ]{ }`, with the only difference that it produces a box by dashed lines. Some applications of `\framebox( )[ ]{ }` and `\dashbox{ }( )[ ]{ }` are shown in Table 10.8 on the next page, where the base coordinates of the boxes are marked by small solid circles for illustrative purpose.

Apart from the above three commands, the `\parbox[pos]{`l_x`}{atext}` command can also be used for printing texts in the `picture` environment. The meanings of the

Table 10.8 Boxed texts in figures using the \framebox( )[ ]{ } and \dashbox{ }( )[ ]{ } commands

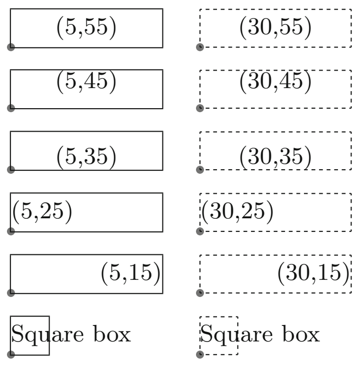
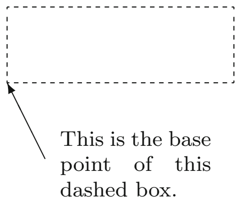
LaTeX input	Output
\setlength{\unitlength}{1mm} \begin{picture}(55,60)(0,0) \put(5,55){\framebox(20,5)[c]{(5,55)}} \put(5,45){\framebox(20,5)[t]{(5,45)}} \put(5,35){\framebox(20,5)[b]{(5,35)}} \put(5,25){\framebox(20,5)[l]{(5,25)}} \put(5,15){\framebox(20,5)[r]{(5,15)}} \put(5,5){\framebox(5,5)[l]{Square box}} % \put(30,55){\dashbox{0.5}(20,5)[c]{(30,55)}} \put(30,45){\dashbox{0.5}(20,5)[t]{(30,45)}} \put(30,35){\dashbox{0.5}(20,5)[b]{(30,35)}} \put(30,25){\dashbox{0.5}(20,5)[l]{(30,25)}} \put(30,15){\dashbox{0.5}(20,5)[r]{(30,15)}} \put(30,5){\dashbox{0.5}(5,5)[l]{Square box}} \end{picture}	

Table 10.9 Multi-line texts in a figure through the \parbox[ ]{ }{ } command

LaTeX input	Output
\setlength{\unitlength}{1mm} \begin{picture}(40,35)(0,0) \put(5,20){\dashbox{0.6}(30,10){}} \put(10,10){\vector(-1,2){5}} \put(12,5){\parbox[b]{2.0cm}{This is the base point of this dashed box.}} \end{picture}	

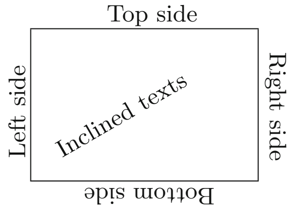
arguments of \parbox[ ]{ }{ } are the same as mentioned above. Like \makebox( )[ ]{ }, \parbox[ ]{ }{ } also does not produce any visible box. As shown in Table 10.9, an advantage of using \parbox[ ]{ }{ } is that a long piece of texts is split over multiple lines, if the size of the box is not sufficient to hold the entire texts in a single line[2].

All the above commands print texts in the horizontal direction. Some texts in a figure may need to be printed in an inclined direction also. The command for the same is \rotatebox{arotate}{atext} defined in the rotating package, which rotates atext by angle arotate (in degree) from the horizontal (a positive value of arotate rotates in the counter-clockwise direction, while a negative value rotates in the clockwise direction). Applications of \rotatebox{ }{ } are shown in Table 10.10 on the next page.

[2] The \parbox[ ]{ }{ } command splits its textual argument over multiple lines, if the size of the box is not sufficient to hold the entire texts in a single line.

Table 10.10 Rotated texts in figures through the `\rotatebox{}{}` command

LaTeX input	Output

`\setlength{\unitlength}{1mm}`
`\begin{picture}(40,37)(0,5)`
`\put(5,15){\framebox(30,20){}}`
`\put(2,18){\rotatebox{90}{Left side}}`
`\put(15,36){\rotatebox{0}{Top side}}`
`\put(36,32){\rotatebox{-90}{Right side}}`
`\put(12,14){\rotatebox{-180}{Bottom side}}`
`\put(8,18){\rotatebox{30}{Inclined texts}}`
`\end{picture}`
	

Note that the commands of Table 17.1 on page 161 can also be used in the **picture** environment through the positioning command `\put(){}` for printing texts in boxes.

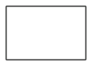
10.6 Compound Figures*

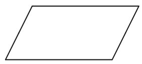
Once commands for drawing different geometric figures are known, they can be combined for forming a compound figure also. Some examples of such figures are given in Table 10.11, where the application of `\multiput()(){}{}` is also shown while drawing parallel lines of the rectangle and parallelogram.

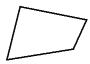
Table 10.11 Compound figures through multiple commands

Name	LaTeX Coding	Figure

Triangle	`\begin{picture}(25,25)(0,0)`
	`\put(0,0){\line(1,0){20}}`
	`\put(0,0){\line(1,2){10}}`
	`\put(20,0){\line(-1,2){10}}`
	`\end{picture}`
		

Rectangle	`\begin{picture}(45,25)(0,0)`
	`\multiput(10,0)(30,0){2}{\line(0,1){20}}`
	`\multiput(10,0)(0,20){2}{\line(1,0){30}}`
	`\end{picture}`
		

Parallelogram	`\begin{picture}(50,40)(0,0)`
	`\multiput(5,5)(10,20){2}{\line(1,0){40}}`
	`\multiput(5,5)(40,0){2}{\line(1,2){10}}`
	`\end{picture}`
		

Quadrilateral	`\begin{picture}(45,25)(0,0)`
	`\curve(10,0, 35,4)`
	`\curve(35,4, 40,15)`
	`\curve(40,15, 15,20)`
	`\curve(15,20, 10,0)`
	`\end{picture}`
		

Decay curve	`\begin{picture}(85,25)(0,0)`
	`\curve(0,20, 5,40, 10,20)`
	`\curve(10,20, 15,0, 20,20)`
	`\curve(20,20, 25,35, 30,20)`
	`\curve(30,20, 35,5, 40,20)`
	`\curve(40,20, 45,30, 50,20)`
	`\curve(50,20, 55,10, 60,20)`
	`\curve(60,20, 65,25, 70,20)`
	`\curve(70,20, 75,15, 80,15)`
	`\end{picture}`
		

Hour 11

Equation Writing I

Mathematical expressions or equations in LaTeX are written in math-mode environments, such as **equation** or **eqnarray**. The math-mode environments are defined in the **amsmath** package, while many mathematical symbols are defined in the **amssymb** package. There exist many more relevant packages, which are stated in Appendix A on page 247.

11.1 Basic Mathematical Notations and Delimiters

Since various mathematical notations are basic tools for writing mathematical expressions, LaTeX commands for some frequently used notations are listed in Table 11.1 on the next page as a quick reference. Note that multi-letter (or multi-digit) superscripts or subscripts must be inserted in **{}**, while a single-letter (or single-digit) can be inserted directly also.

Some basic delimiters, a pair of which acts like parentheses to enclose an expression, are also given in Table 11.2 on the next page (more delimiters are given in Appendix A). To fit automatically around the height of a mathematical expression, the **\left** and **\right** commands may be used before the opening and closing delimiters (**\left** and **\right** are used as a complementary pair). The two delimiters enclosing an expression need not to be similar. For example, **\left(** and **\right]** can be used to enclose an expression in **( ]**. If no delimiter is required in one side of an expression, the **\left.** or **\right.** command, as applicable, may be used. On the other hand, for fixed big-size delimiters (i.e., the size of a delimiter is not auto-adjusted as seen in the right column of Table 11.2), the **\big**, **\Big**, **\bigg** and **\Bigg** commands may be used by appending **l** and **r** for producing opening and closing delimiters respectively, e.g., **\biggl\{** and **\biggr\}** will produce a pair of big-sized curly braces. Note that none of the commands of the forms of **\big**, **\Big**, **\bigg** and **\Bigg** is required to appear in a complementary pair, i.e., either the opening or closing delimiter can also be used alone as shown in the right column of Table 11.2.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_11

Table 11.1 Frequently used mathematical notations (math-mode)

Function	Command with application	Output
Prime	`p'`	p'
Dots	`\dot{x}, \ddot{x}, \dddot{x}, \ddddot{x}`	$\dot{x}, \ddot{x}, \dddot{x}, \ddddot{x}$
Single sub-/super-script	`x_i, x^2`	x_i, x^2
Multiple sub-/super-scripts	`x_{ij}, x^{2k}`	x_{ij}, x^{2k}
Subscript and superscript	`x^{2k}_{ij}` or `x_{ij}^{2k}`	x^{2k}_{ij}
Summation	`\sum, \sum_{i=1}^{20}`	$\sum, \sum_{i=1}^{20}$
Product	`\prod, \prod_{i=1}^{i=20}`	$\prod, \prod_{i=1}^{20}$
Integration	`\int x^2\,dx, \int_a^b xy\,dx`	$\int x^2\,dx, \int_a^b xy\,dx$
Multiple integration	`\iint\limits_s, \iiint\limits_v, \iiiint`	$\iint\limits_s, \iiint\limits_v, \iiiint$
Set of integrations	`\idotsint`	$\idotsint$
Cyclic integration	`\oint`	$\oint$
Fraction	`\frac{x}{y}`	$\frac{x}{y}$
Derivative	`\nabla{f}, \frac{dx}{dy}`	$\nabla f, \frac{dx}{dy}$
Partial derivative	`\frac{\partial{y}}{\partial{x}}`	$\frac{\partial y}{\partial x}$
Root	`\sqrt{x}, \sqrt[5]{xyz}`	$\sqrt{x}, \sqrt[5]{xyz}$
Limit	`\lim_{x\to 0}, \underset{x\to 0}{\lim}`	$\lim_{x\to 0}, \underset{x\to 0}{\lim}$
Exists/not exists	`\exists, \nexists`	$\exists, \nexists$
Modes	`\mod{n^2}, \bmod{n^2}, \pmod{n^2}, \pod{n^2}`	mod n^2, modn^2, (mod n^2), (n^2)
Binomial expression	`\binom{n}{k}`	$\binom{n}{k}$

Table 11.2 Basic delimiters (math-mode)

<table>
<tr><th>Delimiter</th><th>Command</th><th>Delimiter</th><th>Command</th></tr>
<tr><td>$\left(\frac{x}{y} \right)$</td><td>\left(\frac{x}{y} \right)</td><td>$\bigl(\frac{x}{y} \bigr)$</td><td>\bigl(\frac{x}{y} \bigr)</td></tr>
<tr><td>$\left(\frac{x}{y} \right.$</td><td>\left(\frac{x}{y} \right.</td><td>$\Bigl(\frac{x}{y}$</td><td>\Bigl(\frac{x}{y}</td></tr>
<tr><td>$\left. \frac{x}{y} \right)$</td><td>\left. \frac{x}{y} \right)</td><td>$\frac{x}{y} \biggr)$</td><td>\frac{x}{y} \biggr)</td></tr>
<tr><td>$\left\{ \frac{x}{y} \right\}$</td><td>\left\{ \frac{x}{y} \right\}</td><td></td><td></td></tr>
<tr><td>$\left\{ \frac{x}{y} \right.$</td><td>\left\{ \frac{x}{y} \right.</td><td>$\biggl\{ \frac{x}{y} \biggr\}$</td><td>\biggl\{ \frac{x}{y} \biggr\}</td></tr>
<tr><td>$\left. \frac{x}{y} \right\}$</td><td>\left. \frac{x}{y} \right\}</td><td></td><td></td></tr>
<tr><td>$\left[\frac{x}{y} \right]$</td><td>\left[\frac{x}{y} \right]</td><td>$\Biggl(\frac{x}{y} \Biggr)$</td><td>\Biggl(\frac{x}{y} \Biggr)</td></tr>
<tr><td>$\left[\frac{x}{y} \right.$</td><td>\left[\frac{x}{y} \right.</td><td></td><td></td></tr>
<tr><td>$\left. \frac{x}{y} \right]$</td><td>\left. \frac{x}{y} \right]</td><td>$\frac{x}{y} \Biggr\}$</td><td>\frac{x}{y} \Biggr\}</td></tr>
<tr><td>$\left| \frac{x}{y} \right|$</td><td>\left| \frac{x}{y} \right|</td><td></td><td></td></tr>
<tr><td>$\left| \frac{x}{y} \right.$</td><td>\left| \frac{x}{y} \right.</td><td>$\frac{x}{y} \Biggr|$</td><td>\frac{x}{y} \Biggr|</td></tr>
<tr><td>$\left. \frac{x}{y} \right|$</td><td>\left. \frac{x}{y} \right|</td><td></td><td></td></tr>
</table>

11.2 Mathematical Operators

In order to form a mathematical expression, various terms are connected by some operators, which are classified into two categories: binary operators and relation operators. Such basic operators, which are to be obtained in LaTeX through some commands, are given in Tables 11.3 and 11.4 (more operators are given in Appendix A).

Table 11.3 Basic binary operators

Symbol	Command	Symbol	Command	Symbol	Command
$\pm$	\pm	$\diamond$	\diamond	$\wr$	\wr
$\mp$	\mp	$\Diamond$	\Diamond	$\setminus$	\setminus
$\div$	\div	$\triangle$	\triangle	$\amalg$	\amalg
$\times$	\times	$\bigtriangleup$	\bigtriangleup	$\dagger$	\dagger
$\ast$	\ast	$\bigtriangledown$	\bigtriangledown	$\ddagger$	\ddagger
$\star$	\star	$\triangleleft$	\triangleleft	$\bigcirc$	\bigcirc
$\cdot$	\cdot	$\triangleright$	\triangleright	$\bigcap$	\bigcap
$\circ$	\circ	$\lhd$	\lhd	$\bigcup$	\bigcup
$\bullet$	\bullet	$\rhd$	\rhd	$\bigsqcup$	\bigsqcup
$\cap$	\cap	$\unlhd$	\unlhd	$\biguplus$	\biguplus
$\cup$	\cup	$\unrhd$	\unrhd	$\bigvee$	\bigvee
$\sqcap$	\sqcap	$\odot$	\odot	$\bigwedge$	\bigwedge
$\sqcup$	\sqcup	$\oplus$	\oplus	$\bigodot$	\bigodot
$\uplus$	\uplus	$\ominus$	\ominus	$\bigoplus$	\bigoplus
$\vee$	\vee	$\otimes$	\otimes	$\bigotimes$	\bigotimes
$\wedge$	\wedge	$\oslash$	\oslash		

Table 11.4 Basic relation operators

Symbol	Command	Symbol	Command	Symbol	Command
$\leq$	\leq (or, \le)	$\in$	\in	$\not=$	\not=
$\ll$	\ll	$\notin$	\not\in	$\doteq$	\doteq
$\geq$	\geq (or, \ge)	$\ni$	\ni	$\propto$	\propto
$\gg$	\gg	$\vdash$	\vdash	$\vert$	|
$\gtreqqless$	\gtreqqless	$\dashv$	\dashv	$\models$	\models
$\prec$	\prec	$\equiv$	\equiv	$\perp$	\perp
$\preceq$	\preceq	$\not\equiv$	\not\equiv	$\mid$	\mid
$\succ$	\succ	$\sim$	\sim	$\parallel$	\parallel
$\succeq$	\succeq	$\not\sim$	\not\sim	$\not\parallel$	\not\parallel
$\subset$	\subset	$\simeq$	\simeq	$\bowtie$	\bowtie
$\subseteq$	\subseteq	$\asymp$	\asymp	$\Join$	\Join
$\sqsubseteq$	\sqsubseteq	$\approx$	\approx	$\smile$	\smile
$\supset$	\supset	$\not\approx$	\not\approx	$\frown$	\frown
$\supseteq$	\supseteq	$\cong$	\cong	$\not<$	\not<
$\sqsupseteq$	\sqsupseteq	$\neq$	\neq	$\not>$	\not>

11.3 Mathematical Expressions in Text-Mode

A math-mode environment, like **equation** or **enqarray**, prints a mathematical expression in a new line. Sometime a short mathematical notation or expression may need to be printed in running texts also, i.e., in the same line along with texts. For example, in the following statement, one equation and three variables are printed in running texts:

> The equation of an origin-centered circle is $x^2 + y^2 = r^2$, where x and y are the coordinates of a point on the circumference of the circle, and r is its radius.

A mathematical expression, say `amath`, can be inserted in running texts as `$amath$`, `\(amath\)` or `\begin{math}amath\end{math}`, where '`$$`', '`\( \)`' or the **math** environment create math-modes in running texts[1]. A single notation is usually inserted in `$$`, while an expression is inserted in `\( \)` or in the **math** environment (however, all three are applicable in either case). In the above example, accordingly, the equation can be inserted as `\(x^2 + y^2 = r^2\)` or `\begin{math}x^2 + y^2 = r^2\end{math}`, while the variables x, y and r as `$x$`, `$y$` and `$r$` respectively.

11.4 Simple Equations

The very basic math-mode environment for producing an equation is **equation**. Within the environment, an equation can be written as a combination of different mathematical expressions, as shown in Table 11.1, such as summation, power, root or equality. The **equation** environment, an example of which is shown in Table 11.5, is used for

Table 11.5 A simple equation through the **equation** environment

LaTeX input	Output

`\begin{equation}`
`x^2 + y^2 = r^2`
`\label{eq:circ}`
`\end{equation}`
	$x^2 + y^2 = r^2$ (11.1)

inserting a single equation that is printed in a separate center-aligned line. Moreover, the equation is assigned a serial number printed in `( )` on its right hand side. As shown in Table 11.5, an equation can be assigned a unique label-word through the `\label{}` command, which can be used for referring the equation through the `\ref{}` command (similar to `\ref{}`, the `\eqref{}` command may also be used for referring an equation, which automatically puts the serial number of the equation in a pair of parentheses).

[1] A mathematical notation or expression, say `amath`, can be inserted in running texts as `$amath$`, `\(amath\)`, or `\begin{math}amath\end{math}`.

11.4.1 Eliminating Equation Numbering

If an equation is not to be numbered, either the **equation*** environment, or one of the **\nonumber** and **\notag** commands after the equation, may be used. There exist the **displaymath** environment and the **\[\]** mode, which also allow to produce an equation without numbering it. The applications of these provisions are shown in Table 11.6. When an equation is prevented from numbering by any of these approaches, the equation is not counted during numbering its following equations.

Table 11.6 Different approaches for producing equations without numbering

LaTeX input	Output

`\begin{equation}`
`x^2 + y^2 = r^2 \nonumber`
`\end{equation}`
	$x^2 + y^2 = r^2$

`\begin{equation}`
`x^2 + y^2 = r^2 \notag`
`\end{equation}`
	$x^2 + y^2 = r^2$

`\begin{equation*}`
`x^2 + y^2 = r^2`
`\end{equation*}`
	$x^2 + y^2 = r^2$

`\begin{displaymath}`
`x^2 + y^2 = r^2`
`\end{displaymath}`
	$x^2 + y^2 = r^2$

`\[x^2 + y^2 = r^2\]`	$x^2 + y^2 = r^2$

11.4.2 Overwriting Equation Numbering

Opposite to **\notag**, there are **\tag{**`anum`**}** and **\tag*{**`anum`**}** commands allowing to overwrite the numbering of an equation by `anum`, where **\tag{}** prints `anum` in a pair of parentheses and **\tag*{}** prints it without any parenthesis. An example is shown in Table 11.7, where **\tag*{}** is used to refer a label-word by **\ref{}**[2].

Table 11.7 Overwriting equation numbering by the **\tag*{}** command

LaTeX input	Output

`\begin{equation}`
`x^2+y^2=r^2 \tag*{Repeat Eq.~(\ref{eq:circ})}`
`\end{equation}`
	$x^2 + y^2 = r^2$ Repeat Eq. (11.1)

*11.4.3 Changing Printing Format of Equations**

As seen in Table 11.5, by default an equation is printed in a center-aligned new line and it is assigned a serial number in `()` on its right side. All the equations of a doc-

[2]The **\eqref{}** command works in the **\tag{}** command, but not in the **\tag*{}** command.

ument can be made globally left aligned, with some predefined indentation, using **fleqn** as an option to **\documentclass[]{}** (§11.6 on page 110 discusses left alignment of a particular equation only). The predefined indentation under the **fleqn** option can be changed by altering the value of **\mathindent**, e.g., **\setlength{\mathindent}{5mm}** or **\mathindent=0mm**. Such a change may be made in the preamble for global effect, or prior to an equation for local effect. Further, **leqno** may be used as another option to **\documentclass[]{}** for printing equation number on left side. Another noticeable thing in Table 11.5 is that the serial number of the equation is preceded by the chapter number and a period mark. That is, equations in the document-class **book** are numbered chapter-wise. In contrast, an equation in the document-class **article** is assigned its serial number only, i.e., not section-wise (the document-class **article** does not support a chapter). To get the equation numbering section-wise, the following four lines of commands may be included in the preamble (§19.2.5 on page 189 discusses the commands in detail):

```
\makeatletter
\@addtoreset{equation}{section}
\makeatother
\renewcommand{\theequation}{\thesection.\arabic{equation}}
```

Applications of the above mentioned provisions are shown in Table 11.8. In the document-class **article**, tables and figures, which are also by default numbered by their serial numbers only, can be numbered section-wise as above. In that case, **equation** and **\theequation** in **\@addtoreset{}{}** and **\renewcommand{}{}** are to be replaced by **table** and **\thetable** (or, **figure** and **\thefigure**) respectively.

Table 11.8 Changing the standard format for printing and numbering equations

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>\documentclass[fleqn,leqno]{article}
\usepackage{amsmath, amssymb}
\makeatletter
\@addtoreset{equation}{section}
\makeatother
\renewcommand{\theequation}%
                  {\thesection.\arabic{equation}}
%
\begin{document}
\section{First section}
\begin{equation}
x^2+y^2=r^2
\end{equation}
%
\section{Second section}
\mathindent=0mm
\begin{equation}
x^3+y^3=r^3
\end{equation}
\end{document}</pre></td><td>1 First section
(1.1) $x^2 + y^2 = r^2$

2 Second section
(2.1) $x^3 + y^3 = r^3$</td></tr>
</table>

11.5 Array of Equations

The **equation** or **displaymath** environment, or the **\[\]** mode, is used for producing a single equation in a separate line. Sometime a set of simultaneous equations may need to be produced in an array form (one below another). LaTeX provides a number of special environments for producing an array of equations together, instead of producing each equation by a separate **equation** or **displaymath** environment, or **\[\]** mode. Some of such environments, along with their alignment structures, are given in Table 11.9. In these environments, an equation, except the last one of an array, is

Table 11.9 Array of equations producing environments and their alignment structures

Environment	Alignment structure
gather and **gather***	Gather equations without alignment.
eqnarray and **eqnarray***	Allow alignment about a single place only.
align and **align***	Allow alignment about a single place only.
alignat and **alignat***	Allow alignment about multiple places.
xalignat and **xxalignat**	Allow alignment about multiple places. Columns and margins are equally spaced in **xalignat**, while margin spacing is ignored in **xxalignat**.
array	Allows alignment at multiple places. It is to be nested in a math-mode, say the **equation** or **displaymath** environment or the **\[\]**-mode.

terminated by ****. The **\\[**`vsize`**]** command can also be used for providing extra `vsize` vertical space between two equations. The **\displaybreak[**`adigit`**]** command may also be used just before **** as the page breaking instruction after the current equation, with optional `adigit` value of 0–4, where 0 means the provision for breaking and 4 means the must breaking. On the other hand, the **\intertext{**`atext`**}** command after **** allows to insert a few lines of texts (i.e., `atext`) in between two equations maintaining their alignments.

The patterns of aligning the equations of an array differ from environment to environment. The **eqnarray** and **eqnarray*** environments enclose the aligning place by a pair of **&** sign, e.g., **&**=**&** for aligning about the '=' sign, or simply **&&** for aligning about an empty space. The **align** and **align*** environments use a single **&** on the left side of the aligning place, e.g., **&**= for aligning about the '=' sign. Similarly, the **alignat**, **alignat***, **xalignat** and **xxalignat** environments (which allow alignment at multiple places) also use a single **&** on the left side of an aligning place, but with a provision for ending the current aligning place by another **&** before starting the alignment at the next place, e.g., in `x`**&**+**&**`y`**&**=`z`, the first **&** makes an alignment about the '+' sign, which is ended by the next **&** before starting the second alignment about the '=' sign. The **alignat**, **alignat***, **xalignat** and **xxalignat** environments take the number of aligning places as a mandatory argument, e.g., **\begin{alignat}{**`m`**}** with $\mathtt{m} = \frac{\mathtt{n}}{2} + 1$ if `n` is even and $\mathtt{m} = \frac{\mathtt{n}+1}{2}$ if `n` is odd, where `n` is the number of **&** to be used in an equation. Note that the alignment about a place, under these environments, internally splits an equation at this place into two parts of **rl**-alignment, i.e., the left side part is right aligned and the right side part is left aligned. On the other hand, the aligning process in the **array** environment is quite different. Similar to the **tabular** environment used for preparing a table (refer §7.1 on page 59), the **array** environment creates aligning places through mandatory options of **l** for left alignment, **c** for centered and **r** for right alignment, e.g., \begin{array}{rl} for right aligning the left portion (due to r) and left aligning the right portion (due to l).

The starred forms of the environments (including xxalignat which acts like the starred form of xalignat) ignore the numbering to any equation, while their non-starred forms (including xalignat) assign an individual serial number to each equation. If required, numbering to any equation of an array can be eliminated using the \nonumber or \notag command as explained in §11.4.1 on page 105. On the other hand, the array environment also ignores the numbering, but the entire array of equations will be assigned a single serial number if the array environment is nested inside an equation environment.

Table 11.10 shows, through the same array of three equations, some applications of the environments given in Table 11.9, along with those of other commands discussed above (except \displaybreak). Since the gather environment just gathers an array of equations without any alignment, no & is used in this environment. The

Table 11.10 Array of equations in different forms

LaTeX input	Output

`\begin{gather}`
`5x+ 2y = x+ 2z+ 3\\`
`130x+ 4z = y+ 2\\`
`43y+ 57z = 20x+ 99`
`\end{gather}`
	$5x+2y=x+2z+3$ (11.2)
	$130x+4z=y+2$ (11.3)
	$43y+57z=20x+99$ (11.4)

`\begin{eqnarray}`
`5x+2y &=& x+2z+3 \label{eqn1}\\`
`130x+4z &=& y+2 \nonumber \\`
`43y+57z &=& 20x+99 \label{eqn3}`
`\end{eqnarray}`
	$5x+2y = x+2z+3$ (11.5)
	$130x+4z = y+2$
	$43y+57z = 20x+99$ (11.6)

`\begin{align}`
`5x+2y &= x+2z+3 \tag{See \eqref{eqn1}}\\`
`130x+4z &= y+2 \label{align2}\\[3mm]`
`43y+57z &= 20x+99 \notag`
`\end{align}`
	$5x+2y=x+2z+3$ (See (11.5))
	$130x+4z=y+2$ (11.7)
	
	$43y+57z=20x+99$

`\begin{alignat*}{7}`
`5x&+& 2y&& &=& x&+& && 2z&+& 3\\`
`\intertext{Please notice the alignment …}`
`130x&+& && 4z &=& && y&+& && 2\\`
`&& 43y&+& 57z &=& 20x&+& && && 99`
`\end{alignat*}`
	$5x+\ 2y \quad = \quad x+ \quad 2z+\ 3$
	Please notice the alignment made about each '+' and '=' signs of these equations.
	$130x+ \quad 4z = \quad y+ \quad 2$
	$43y+57z = 20x+ \quad 99$

`\begin{xxalignat}{7}`
`5x&+& 2y&& &=& x&+& && 2z&+& 3\\`
`130x&+& && 4z &=& && y&+& && 2\\`
`&& 43y&+& 57z &=& 20x&+& && && 99`
`\end{xxalignat}`
	$5x+ \quad 2y \quad = \quad x+ \quad 2z+ \quad 3$
	$130x+ \quad 4z = \quad y+ \quad 2$
	$43y+ \quad 57z = \quad 20x+ \quad 99$

`\begin{equation}`
`\left.\begin{array}{*{13}{@{}r@{}}}`
`5x&+& 2y&& &=& x&+& && 2z&+& 3\\`
`130x&+& && 4z &=& && y&+& && 2\\`
`&& 43y&+& 57z &=& 20x&+& && && 99`
`\end{array}\right}`
`\end{equation}`
	$\left.\begin{array}{r} 5x+\ 2y = x+ \quad 2z+\ 3\\ 130x+ \quad 4z= \quad y+ \quad 2\\ 43y+57z=20x \quad 99 \end{array}\right\}$ (11.8)

second equation of the **eqnarray** environment is prevented from numbering by using **\nonumber** before terminating it by ****. For the same purpose, **\notag** is used before terminating the last equation of the **align** environment. Moreover, the first equation of the **align** environment is not numbered but referred to another equation through **\tag{}**, while a gap of 3 mm is created above its last equation by terminating the previous equation by **\\[**3**mm]**. On the other hand, a note is produced, through **\intertext{}**, after the first equation of the **alignat*** environment (**\intertext{}** cannot be used in the **array** environment).

In the **array** environment in Table 11.10, each term and operator, totaling 13 in number, is right aligned separately, which are done by creating 13 number of right aligned places (columns) through **\begin{array}{*{**13**}{r}}** (refer §7.6 on page 64 for detail). The **array** environment allows to enclose an array by a pair of delimiters, which is demonstrated in Table 11.10, where the entire environment is enclosed by **\left.** and **\right\}** for producing a curly brace on the right side of the array. Alternatively, the same effect can be produced by enclosing the mandatory argument of the environment by a pair of delimiters (this provision is defined in the **delarray** package), e.g., it could be created in Table 11.10 as **\begin{array}.{*{**13**}{r}}\}** instead of enclosing the entire environment by **\left.** and **\right\}**. On the other hand, the **array** environment has optional provision for vertical alignment also, which is executed as **\begin{array}[**`valign`**]{}**, where the permissible values of `valign` are **t** for top alignment, **c** for center alignment and **b** for bottom alignment. Further, as adjusting column width in tables discussed in §7.5 on page 63, the blank space between two columns in the **array** environment can be changed either by using **@{}** (as shown in the last example in Table 11.10) or by changing the value of **\arraycolsep** (default is 5 pt), e.g., **\setlength{\arraycolsep}{**1**mm}**. Moreover, any row, column or entry in the **array** environment can be colored in the same way discussed in §7.9 on page 68 for those of a table.

Notice in Table 11.10 that the **eqnarray** environment leaves excess blank space around an aligned place, while other environments of Table 11.9 maintain comparatively better spacing. Further, alignment at a single place may not always be preferred, e.g., the case of the array of equations considered in Table 11.10, where all the variables (`x`, `y` and `z`) may need to be aligned. However, the choice of an environment is up to a user.

Each of the equations of a numbered array, produced by a non-starred environment given in Table 11.9 (excluding **xxalignat** and **array**), can be labeled and referred individually through **\label{}** and **\ref{}**, respectively. As shown in Table 11.10, an equation of an array is to be labeled before terminating it, i.e., before the line-break command ****.

11.6 Left Aligning an Equation*

By default an equation is printed in a center-aligned new line. The **fleqn** option to **\documentclass[]{}** acts globally to make all equations of a document left aligned. In contrast, the **flalign** environment (or the **flalign*** form for unnumbered equations) can be used for left aligning a particular equation only.

The **flalign** environment allows the use of two **&** for internally aligning an array of equations about a particular place, out of which one **&** must be at the end of an equation (refer §11.5 on page 107 for detail of using **&** in an array of equations). If no internal alignment of an array of equations is required about any particular place (or in the case of a single equation), the first **&** may be inserted either at the start or at the end (together with the second **&**) of an equation as per requirement.

Some applications of the **flalign** and **flalign*** environments are shown in Table 11.11, where the locational effect of the first **&** is noticeable. In the case of a single

Table 11.11 Left aligned equations through the **flalign** and **flalign*** environments

LaTeX input	Output

`\begin{flalign*}`
`& x^2+y^2 = r^2 &`
`\end{flalign*}`
	$x^2 + y^2 = r^2$

`\begin{flalign*}`
`x^2+y^2 = r^2 &&`
`\end{flalign*}`
	$x^2 + y^2 = r^2$

`\begin{flalign}`
`& 2x+3y = 6-z & \\`
`&      z = 1   &`
`\end{flalign}`
	$2x + 3y = 6 - z$ (11.9)
	$z = 1$ (11.10)

`\begin{flalign}`
`2x+3y = 6-z && \\`
`    z = 1   &&`
`\end{flalign}`
	$2x + 3y = 6 - z$ (11.11)
	$z = 1$ (11.12)

`\begin{flalign}`
`2x+3y &= 6-z & \\`
`    z &= 1   &`
`\end{flalign}`
	$2x + 3y = 6 - z$ (11.13)
	$z = 1$ (11.14)

`\begin{flalign}`
`\left.\begin{array}{*{13}{@{}r@{}}}`
`  5x&+&  2y&&    &=&   x&+&  &&2z&+&  3\\`
`130x&+&    && 4z &=&    &&y&+&   && 2\\`
`     && 43y&+&57z &=&20x&+&  &&   &&99`
`\end{array}\right}`
`&&`
`\end{flalign}`
	$\left.\begin{array}{rrrrrrrrrrrrr} 5x+ & 2y & & = & x+ & & 2z+ & 3 \\ 130x+ & & 4z= & & & y+ & & 2 \\ & 43y+57z=20x & & & & & & 99 \end{array}\right\}$ (11.15)

equation, as shown in the first two examples, the location of the first **&** does not effect the presentation of the equation. However, the location of the first **&** matters in an array of equations, which can be noticed in the third, fourth and fifth examples in Table 11.11. The last example in Table 11.11 is an interesting one (it is the last example of Table 11.10). Since internal alignment of the array of equations is required about more than one place, the array is first produced through an **array** environment. Then the **array** environment, followed by **&&**, is put in a **flalign** environment for left aligning the entire array.

11.7 Sub-numbering a Set of Equations*

It is seen in §11.5 that the non-starred environments of Table 11.9 (excluding **xxalignat** and **array**) assign an individual serial number to each of a set of equations. Instead of such individual numbering, sometime a set of equations may be preferred to be sub-numbered under a main number, e.g., (3a), (3b), (3c), etc. Such sub-numbering can be obtained by nesting the equation generating environments (like **equation**, **eqnarray** or **align**) in the **subequations** environment. Such an example is shown in Table 11.12,

Table 11.12 Sub-numbering a set of equations

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>
\begin{subequations}
%
\begin{equation}
5x+2y = 2z+3 \label{eq1}
\end{equation}
\begin{equation}
13x = y+z+2 \label{eq2}
\end{equation}
Eqs.~\eqref{eq1} and \eqref{eq2} ...
%
\begin{eqnarray}
 5x+2y &=& 2z+3 \label{arr1}\\
 13x   &=& y+z+2 \label{arr2}
\end{eqnarray}
The same equations are arranged ...
%
\begin{align}
 5x+2y &= 2z+3 \label{algn1}\\
 13x   &= y+z+2 \label{algn2}
\end{align}
The equations are reproduced in ...
%
\label{sub_arrys}
\end{subequations}
%
Eq.~\eqref{sub_arrys} illustrates ...
</pre></td><td>

$$5x + 2y = 2z + 3 \qquad (11.16a)$$
$$13x = y + z + 2 \qquad (11.16b)$$

Eqs. (11.16a) and (11.16b) are produced by two separate **equation** environments.

$$5x + 2y \;=\; 2z + 3 \qquad (11.16c)$$
$$13x \;=\; y + z + 2 \qquad (11.16d)$$

The same equations are arranged in Eqs. (11.16c) and (11.16d) through an **eqnarray** environment.

$$5x + 2y = 2z + 3 \qquad (11.16e)$$
$$13x = y + 3z + 2 \qquad (11.16f)$$

The equations are reproduced in another way in Eqs. (11.16e) and (11.16f) through an **align** environment.

Eq. (11.16) illustrates the sub-numbering of a set of equations produced by different environments.

</td></tr>
</table>

where a set of two equations is produced in three different ways under a single **subequations** environment. In the first case, the two equations are produced through two **equation** environments. In the second and third cases, these are produced through **eqnarray** and **align** environments, respectively. Each equation under the **subequations** environment can be labeled by a unique label-word, as well as the entire set of equations by a single label-word, which can be used for independently referring any or the entire set of equations as shown in Table 11.12. It is also shown in Table 11.12 that normal texts can also be inserted between two equation generating environments under the same **subequations** environment.

Hour 12

Equation Writing II

Writing of basic equations is discussed in Hour 11. Some processes for writing complicated equations are presented here, including the use of mathematical symbols for which special commands are required (such commands are listed in Appendix A on page 247).

12.1 Texts and Blank Space in Math-Mode

Every character in math-mode is treated as a variable and it is printed in math-mode (similar to italic fonts) without any gap between two characters. Normal texts with usual inter-word spacing can be printed in math-mode through the **\mbox{}**, **\text{}** and **\mathrm{}** commands. The commands and fonts discussed in §2.2 on page 11 are also permitted for printing normal texts in math-mode. On the other hand, **~**, **\,**, **\quad**, **\qquad**, **\enspace** and **\hspace{}** can be used in math-mode for generating blank space of different sizes.

Applications of some of the above commands are shown in Table 12.1 on the next page (the same can be found in other sections of the book). In the first example, **\mbox{}** (function of **\text{}** is the same) is used for printing in-line normal texts and **\mathrm{}** for printing a superscript in normal fonts, while **\enspace** is used for maintaining some gap prior to the period mark (the arguments of **\mbox{}** and **\text{}** are in text-mode, while that of **\mathrm{}** is in math-mode)[1]. In the second example in Table 12.1, an array of equations is produced through the **array** environment with alignment at three places. In this example, **\mbox{\boldmath{$**x**$}}** is used for printing x as $\boldsymbol{x}$ (just **\boldmath{**x**}** would print x as x).

[1]The **\mbox{}**, **\text{}** and **\mathrm{}** commands can be used for producing normal texts with usual inter-word spacing in math-modes, where **\mbox{}** and **\text{}** process their arguments in text-mode while **\mathrm{}** processes its argument in math-mode.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_12

Table 12.1 Normal texts and gap in math-mode

LaTeX input	Output

`\begin{equation*}`
`\mbox{Updated value}\quad x =`
`x^\mathrm{low} + yd \enspace .`
`\end{equation*}`
	$$\mbox{Updated value}\quad x = x^\mathrm{low} + yd \enspace .$$

`\begin{equation*}`
`\begin{array}{lll}`
`\mbox{Minimize}`
`& f(\mbox{\boldmath{$x$}}) &\\`
`\mbox{Subject to}`
`& g_i(\mbox{\boldmath{$x$}}) \leq 0~; &`
`i=1,\ldots,m\\`
`& h_k(\mbox{\boldmath{$x$}}) = 0~; &`
`k=1,\ldots,p\\`
`& x_j \geq 0~; & j=1,\ldots,n`
`\end{array}`
`\end{equation*}`
	$$\begin{array}{lll} \mbox{Minimize} & f(\boldsymbol{x}) & \\ \mbox{Subject to} & g_i(\boldsymbol{x}) \leq 0~; & i=1,\ldots,m \\ & h_k(\boldsymbol{x}) = 0~; & k=1,\ldots,p \\ & x_j \geq 0~; & j=1,\ldots,n \end{array}$$

12.2 Conditional Expression

A conditional expression is a case, where a parameter or an expression may take different values in different circumstances. Table 12.2 presents three approaches for

Table 12.2 Conditional mathematical expressions in different forms

LaTeX input	Output

`\begin{equation}`
`\sigma(x) = \begin{cases}`
`e^{\phi␣xy} \sqrt{x}~,`
`& \text{if}~ x\geq 0\\`
`0~, & \text{otherwise.}`
`\end{cases}`
`\end{equation}`
	$$\sigma(x) = \begin{cases} e^{\phi xy} \sqrt{x}~, & \text{if}~ x\geq 0 \\ 0~, & \text{otherwise.} \end{cases} \quad (12.1)$$

`\begin{equation}`
`\sigma(x) = \left\{\begin{array}{rl}`
`e^{\phi␣xy} \sqrt{x}~,`
`& \text{if}~ x\geq 0\\`
`0~, & \text{otherwise.}`
`\end{array}\right.`
`\end{equation}`
	$$\sigma(x) = \left\{\begin{array}{rl} e^{\phi xy} \sqrt{x}~, & \text{if}~ x\geq 0 \\ 0~, & \text{otherwise.} \end{array}\right. \quad (12.2)$$

`\begin{eqnarray}`
`\sigma(x) &=& e^{\phi␣xy} \sqrt{x}~,\quad`
`\text{if}~ x\geq 0\\`
`&=& 0~,\quad \text{otherwise.}`
`\end{eqnarray}`
	$$\sigma(x) = e^{\phi xy} \sqrt{x}~,\quad \text{if}~ x\geq 0 \quad (12.3)$$ $$= 0~,\quad \text{otherwise.} \quad (12.4)$$

printing such conditional expressions. The first approach uses the **cases** environment inside an **equation** environment, in which conditions are preceded by **&** sign for aligning them. In the **cases** environment, all the conditions are enclosed by a left-hand curly brace, and the entire conditional expression is assigned a single serial number. On the other hand, the **array** environment, applied in the second approach in Table 12.2, is more flexible than the **cases** environment. In the **array** environment, different types of auto-sized delimiters can be obtained on either side of an expression. Moreover, different parts of the expression can be aligned differently. Like the **cases** environment, the **array** environment also can assign a serial number to a set of expressions, if it is nested in a math-mode, such as the **equation** environment. If all the conditions of an expression are to be numbered independently, the **enqarray** environment may be used as shown in Table 12.2 as the third approach for printing the given two conditions by aligning them about the '=' sign (similar effect can be obtained by using other environments given in Table 11.9 on page 107). Note that a gap is to be maintained between **\phi** and `xy` in Table 12.2 (refer §1.5.1 on page 5 for detail).

12.3 Evaluation of Functional Values

Evaluation of a function is an important part of mathematics. Table 12.3 shows an

Table 12.3 Evaluation of a function for a given value

LaTeX input	Output

`\begin{equation*}`
`\left. f\left(\frac{x+1}{2}+2\right) \right|_{x=0}=2.5`
`\end{equation*}`
	$\left. f\left(\frac{x+1}{2}+2\right) \right\|_{x=0}=2.5$

example of evaluating $f(x)$ at $x = 0$, where a vertical line of auto-adjusted height is first produced by using the set of **\left.** and **\right|** commands around the functional expression. Then, the given value ($\mathtt{x} = \mathtt{0}$) is printed as the suffix of **|** through `|_{x=0}`.

12.4 Splitting an Equation into Multiple Lines*

It is stated in §11.4 on page 104 that the **equation** or **displaymath** environment, or even the **\[\]**-mode, produces a single equation in a single new line. If an equation is long enough to accommodate in a single line, it can be split into multiple lines through the **multline** environment or its starred form **multline***. In these environments, the first split line is left aligned, the last one is right aligned and all other intermediate lines are centered as shown in Table 12.4 on the next page. The **multline** environment assigns a serial number to the equation, while the **multline*** ignores its numbering.

Table 12.4 Splitting an equation into multiple lines through the **multline** environment

LaTeX input	Output

`\begin{multline}`
`5x_1 + 2x_2 + 3x_3 -\\`
`x_4 - 4x_5 + 5x_6 +\\`
`7x_7 + 3x_8 - 6x_9 -\\`
`2x_{10} - 5x_{11} = 7634`
`\end{multline}`
	$$\begin{multline}5x_1 + 2x_2 + 3x_3 -\\ x_4 - 4x_5 + 5x_6 +\\ 7x_7 + 3x_8 - 6x_9 -\\ 2x_{10} - 5x_{11} = 7634 \tag{12.5}\end{multline}$$

Another environment for splitting a long equation into multiple lines is **split**. Like the **array** environment, the **split** environment is also nested in another math-mode, such as the **equation** or **displaymath** environment or **\[\]** mode. An application of this environment is shown in Table 12.5. By default the **split** environment makes all the

Table 12.5 Splitting an equation into multiple lines through the **split** environment

LaTeX input	Output

`\begin{equation}`
`  \begin{split}`
`    f(x,y)=h \biggl[ & \frac{1}{2}(x+y)+x^2+y^3\\`
`                    & +\frac{1}{3}z^2\biggr]`
`  \end{split}`
`\end{equation}`
	$$\begin{split} f(x,y)=h \biggl[& \frac{1}{2}(x+y)+x^2+y^3\\ & +\frac{1}{3}z^2\biggr] \end{split} \tag{12.6}$$

lines right aligned. Alignment about another place can be obtained by using an **&** at that place, as shown in Table 12.5 by aligning after the left-hand square bracket. Note that since **\left** and **\right** (used for obtaining auto-sized delimiters) appear as a matching pair, they cannot be split into two lines. In such cases, instead of **\left** and **\right**, the commands of the forms of **\big**, **\Big**, **\bigg** and **\Bigg** (refer §11.1 on page 101 for detail) may be used as shown in Table 12.5, where **\biggl** and **\biggr** are applied for generating two big-sized square brackets in two split lines.

As shown in Table 12.6, the **split** environment can be used in mathematical analysis also, where the multiple lines are aligned about the '=' sign.

Table 12.6 Mathematical analysis through the **split** environment

LaTeX input	Output

`\begin{equation*}`
`  \begin{split}`
`    f(x) &= x^3 + 2x^2 - 5x + 10\\`
`         &= (2)^3 + 2(2)^2 - 5(2) + 10\\`
`         &= 16`
`  \end{split}`
`\end{equation*}`
	$$\begin{split} f(x) &= x^3 + 2x^2 - 5x + 10\\ &= (2)^3 + 2(2)^2 - 5(2) + 10\\ &= 16 \end{split}$$

The **\lefteqn{**`fline`**}** command can also be used in the **eqnarray** environment for splitting a long equation into multiple lines, where `fline` is the first split line. The first example in Table 12.7 shows the use of **\lefteqn{}** for reproducing the

Table 12.7 Splitting an equation into multiple lines through the **eqnarray** environment

LaTeX input	Output

`\begin{eqnarray}`
`\lefteqn{f(x,y)=h\biggl[\frac{1}{2}(x+y)+x^2+y^3}`
`\nonumber\\`
`&& +\frac{1}{3}z^2\biggr]`
`\end{eqnarray}`
	$f(x,y) = h\biggl[\frac{1}{2}(x+y) + x^2 + y^3$
	$+\frac{1}{3}z^2\biggr] \quad (12.7)$

`\begin{eqnarray}`
`\lefteqn{f(x,y)=h\biggl[\frac{1}{2}(x+y)+x^2+y^3}`
`\hspace{9mm}\nonumber\\`
`&& +\frac{1}{3}z^2\biggr]`
`\end{eqnarray}`
	$f(x,y) = h\biggl[\frac{1}{2}(x+y) + x^2 + y^3$
	$+\frac{1}{3}z^2\biggr] \quad (12.8)$

equation of Table 12.5, where the second line of the equation is preceded by **&&** for printing it with default left indentation. As shown in the second example, the indentation of the following lines can be increased by using **\hspace{}** after **\lefteqn{}** in the first part. Since the **enqarray** environment assigns a default serial number to each equation, **\nonumber** is used in the first split lines in Table 12.7 for ignoring their numbering.

12.5 Vector and Matrix

The direct LaTeX environments for producing a matrix include **matrix**, **pmatrix**, **bmatrix**, **vmatrix** and **Vmatrix**. These environments work for a vector also, where it is to be inserted as a single column or single row matrix. The environments are to be nested in a math-mode, such as the **equation** environment. Like in the **tabular** environment used for producing a table (refer §7.1 on page 59), the entries of two columns in a matrix generating environment is separated by a **&** sign and a row other than the last one is terminated by the line break command ****. The **matrix** environment produces a matrix without any delimiter, while the delimiters in the **pmatrix**, **bmatrix**, **vmatrix** and **Vmatrix** environments are (), [ ], | | and ‖ ‖, respectively. Applications of these five environments are shown in Table 12.8 on the next page. Apart from these, there is another environment, **smallmatrix**, which is useful for producing a small matrix or vector in text-mode, e.g., **$\left[\begin{smallmatrix}**`a`**&**`b`**\\**`c`**&**`d`**\end{smallmatrix}\right]$** will produce $\left[\begin{smallmatrix}a&b\\c&d\end{smallmatrix}\right]$.

The above matrix generating environments have some limitations. Firstly, a matrix only up to a maximum of 10 columns can be generated. For a higher order matrix, the value of the column counter **MaxMatrixCols** is to be reset, e.g., **\setcounter{MaxMatrixCols}{**`15`**}** for producing a matrix up to a maximum of 15

Table 12.8 Matrices and vectors through direct environments

LaTeX input	Output
`\begin{equation*}` `\begin{matrix} 3a & b \\ c & 5d \end{matrix}` `\end{equation*}`	$\begin{matrix} 3a & b \\ c & 5d \end{matrix}$
`\begin{equation*}` `\begin{pmatrix} x_1 \\ x_2 + 7 \end{pmatrix}` `\end{equation*}`	$\begin{pmatrix} x_1 \\ x_2 + 7 \end{pmatrix}$
`\begin{equation*}` `\begin{bmatrix} 1-y & 0 \\ 0 & 1-y \end{bmatrix}` `\end{equation*}`	$\begin{bmatrix} 1-y & 0 \\ 0 & 1-y \end{bmatrix}$
`\begin{equation*}` `\begin{vmatrix} 50 & 0 \\ 0 & 75 \end{vmatrix}` `\end{equation*}`	$\begin{vmatrix} 50 & 0 \\ 0 & 75 \end{vmatrix}$
`\begin{equation*}` `\begin{Vmatrix} \lambda_1 \\ \lambda_2 + 9 \end{Vmatrix}` `\end{equation*}`	$\begin{Vmatrix} \lambda_1 \\ \lambda_2 + 9 \end{Vmatrix}$

columns. Secondly, the environments do not have any column formatting option, but all entries are made center aligned. Moreover, as seen in Table 12.8, no one generates a matrix in { }, which is often used particularly in vectors. However, required delimiters can be produced through the **matrix** environment by enclosing it with the delimiters, e.g., **\left\{\begin{matrix}**...**\end{matrix}\right\}** for producing the matrix in { }.

A good alternative to overcome the limitations of the matrix generating environments is to use the **array** environment. For illustration of the environment, Table 12.9

Table 12.9 Matrices and vectors through the **array** environment

LaTeX input	Output
`\begin{equation*}` `\vec{\mbox{\boldmath$x$}} = \left\{ \begin{array}{l}` `x_1\\x_2\\ \vdots\\x_n + k` `\end{array} \right\}` `\end{equation*}`	$\vec{\boldsymbol{x}} = \left\{ \begin{array}{l} x_1\\ x_2\\ \vdots\\ x_n + k \end{array} \right\}$
`\begin{equation*}` `\left[ \begin{array}{rrr}` `33 & 0 & 375\\` `289 & 470 & 8\\` `7 & 14 & 67` `\end{array} \right]` `\end{equation*}`	$\left[\begin{array}{rrr} 33 & 0 & 375\\ 289 & 470 & 8\\ 7 & 14 & 67 \end{array} \right]$

shows an example of a column vector in { } with left aligned elements, and a matrix in [] with right aligned elements. Further, as the application of the **array** environment

Table 12.10 Matrix and vector mixed expression through the **array** environment

LaTeX input	Output

`\begin{equation*}`
`\left[\begin{array}{cccc}`
`k_{11} & k_{12} & \ldots & k_{1n}\\`
`k_{21} & k_{22} & \ldots & k_{2n}\\`
`\hdotsfor{4}\\`
`k_{n1} & k_{n2} & \ldots & k_{nn}`
`\end{array}\right]`
`%`
`\left\{\begin{array}{c}`
`x_1\\x_2\\ \hdotsfor{1}\\x_n`
`\end{array}\right\} =`
`%`
`\left\{\begin{array}{l}`
`f_1 + a\\f_2\\ \hdotsfor{1}\\f_n + c`
`\end{array}\right\}`
`\end{equation*}`
	$\left[\begin{array}{cccc} k_{11} & k_{12} & \ldots & k_{1n}\\ k_{21} & k_{22} & \ldots & k_{2n}\\ \cdots & \cdots & \cdots & \cdots\\ k_{n1} & k_{n2} & \ldots & k_{nn} \end{array}\right] \left\{\begin{array}{c} x_1\\x_2\\ \cdot\\x_n \end{array}\right\} = \left\{\begin{array}{l} f_1 + a\\f_2\\ \quad\ldots\\f_n + c \end{array}\right\}$

for generating matrices and vectors, an expression is shown in Table 12.10, which contains a matrix and two vectors. Hence, the expression is inserted in three parts using three individual **array** environments, the first one is for the matrix and the remaining two are for the two vectors. Note that there should not be any line break command or blank line after an **array** environment, otherwise the contents of the next **array** environment will be printed in the following line, instead of in the same line. The `\hdotsfor[aspace]{n}` command draws a horizontal dotted line, in the **array** or any other matrix generating environment, over `n` number of columns with `aspace` as the optional dot spacing, e.g., `\hdotsfor[1.5]{5}` for a line over 5 columns with 1.5 spacing between two dots.

12.6 Overlining and Underlining

Sometime an expression can be presented by putting a line over a term, instead of enclosing it in a pair of delimiters. Such an overlining is done through the `\overline{}` command, an example of which is shown in Table 12.11. Similarly the

Table 12.11 Mathematical expression overlined through the `\overline{}` command

LaTeX input	Output

`\begin{equation*}`
`S = \frac{n}{2}\left(2a+\overline{n-1}d\right)`
`\end{equation*}`
	$S = \frac{n}{2}\left(2a+\overline{n-1}d\right)$

`\underline{}` command can be used for putting a line under a term. On the other hand, the `\overbrace{}` or `\underbrace{}` command can be used for putting a brace over or under a term. Moreover, a note can also be placed over or under such a brace. Examples of both the cases are shown in Table 12.12.

Table 12.12 Mathematical expression with over and under braces

LaTeX input	Output

`\begin{equation*}`
`A = \overbrace{a_{11}+a_{12}+a_{13}}`
`+ \underbrace{a_{21}+a_{22}+a_{23}}`
`\end{equation*}`
	$A = \overbrace{a_{11}+a_{12}+a_{13}} + \underbrace{a_{21}+a_{22}+a_{23}}$

`\begin{equation*}`
`A = \overbrace{a_{11}+a_{12}+a_{13}}^I`
`+ \underbrace{a_{21}+a_{22}+a_{23}}_{II}`
`\end{equation*}`
	$A = \overbrace{a_{11}+a_{12}+a_{13}}^I + \underbrace{a_{21}+a_{22}+a_{23}}_{II}$

A note over an overbrace is put as the superscript to `\overbrace{}`. Similarly, a note under an underbrace is put as the subscript to `\underbrace{}`.

12.7 Stacking Terms*

In many applications, like in chemical reactions, two terms often need to be stacked (to put one above another), where the upper term is usually some texts and lower one is a symbol covering the upper term. Generally stacking is done through the `\stackrel{aup}{alow}`, where `aup` is the upper term and `alow` is the lower one. Some applications of this command are shown in Table 12.13, where arrow symbols are generated through fixed-length based direct arrow commands.

Table 12.13 Stacking a mathematical term with an arrow of fixed length

LaTeX input	Output

`\begin{equation*}`
`A \stackrel{a}{\rightarrow} B`
`\end{equation*}`
	$A \stackrel{a}{\rightarrow} B$

`\begin{equation*}`
`\mathrm{2Na + D_2}`
`\stackrel{\mathrm{heat}}{\longrightarrow}`
`\mathrm{2NaD}`
`\end{equation*}`
	$\mathrm{2Na + D_2} \stackrel{\mathrm{heat}}{\longrightarrow} \mathrm{2NaD}$

`\begin{equation*}`
`\mathrm{NH_3}`
`\stackrel{\mathrm{D_2}}{\rightleftharpoons}`
`\mathrm{NH_2D}`
`\stackrel{\mathrm{D_2}}{\rightleftharpoons}`
`\mathrm{ND_3}`
`\end{equation*}`
	$\mathrm{NH_3} \stackrel{\mathrm{D_2}}{\rightleftharpoons} \mathrm{NH_2D} \stackrel{\mathrm{D_2}}{\rightleftharpoons} \mathrm{ND_3}$

The stacking under `\stackrel{}{}` may look odd if the length of the term is too long or short in comparison to that of the covering symbol. Such problems may arise with arrows whose lengths are predefined. In that situation, stacking may be done through commands like \overleftarrow{}, \overrightarrow{}, \underleftarrow{} and \underrightarrow{}, which produce arrows of flexible lengths to cover their arguments. For proper presentation, \overleftarrow{} and \overrightarrow{}, which produce arrows on top, should be used as subscripts. On the other hand, \underleftarrow{} and \underrightarrow{}, which produce arrows at bottom, should be used as superscripts. Table 12.14 shows applications of \overrightarrow{} and \underrightarrow{} on the

Table 12.14 Stacking a mathematical term with an over or under arrow of flexible length

LaTeX input	Output
\begin{equation*} \mathrm{H_2 + D_2} {}_{\overrightarrow{~\mathrm{600\text{--}750\,^oC}~}} \mathrm{2HD} \end{equation*}	$\mathrm{H_2 + D_2}{}_{\overrightarrow{~\mathrm{600\text{--}750\,^oC}~}}\mathrm{2HD}$
\begin{equation*} \mathrm{H_2 + D_2} {}^{\underrightarrow{~\mathrm{600\text{--}750\,^oC}~}} \mathrm{2HD} \end{equation*}	$\mathrm{H_2 + D_2}{}^{\underrightarrow{~\mathrm{600\text{--}750\,^oC}~}}\mathrm{2HD}$

same example, where the commands are inserted, respectively, as the subscript and superscript to an empty character, i.e., to {} (it can be subscribed to the previous term also). Since \overrightarrow{} and \underrightarrow{} produce arrows of lengths equal to those of their arguments, ~ is added on either side of their arguments for producing arrows of slightly bigger lengths. Further, since the dash producing command '--' does not work in math-mode, it is inserted through the \text{} command.

LaTeX also provides the \overset{atop}{abot} and \underset{abot}{atop} commands for stacking two terms, where atop is produced on the top of abot, e.g., \overset{a}{X}, \underset{b}{X} and \overset{a}{\underset{b}{X}} will produce $\overset{a}{X}$, $\underset{b}{X}$ and $\overset{a}{\underset{b}{X}}$ respectively. Notice the expressions in Table 12.14 – both of \overrightarrow{} and \underrightarrow{} print their arguments on one side of the arrows. If terms on both sides of an arrow are required, either the pair of \underset{}{} and \underrightarrow{}, or \overset{}{} and \overrightarrow{} may be used. Applications of both the pairs are shown in Table 12.15 on the next page, where it is to be noticed that the smaller stacking term is taken as the first argument of \underset{}{} or \overset{}{} (otherwise an arrow of a smaller length will be produced).

The \overset{}{} and \underset{}{} commands can also be used for printing ranges or limits of big symbols like $\sum$ and $\prod$. Generally, the ranges of these symbols are inserted as superscripts and subscripts, and these are printed on the top and at bottom of the symbols, e.g., '\sum^n_{i=1} x_i' in most of the math-modes will produce $\displaystyle\sum_{i=1}^{n} x_i$, but $\sum^n_{i=1} x_i$ in text-mode and array environment (i.e., like superscript and subscript on the right side). In such cases, \overset{}{} and \underset{}{} can be used for forcibly printing the ranges on the top and at bottom of a symbol,

Table 12.15 Stacking two mathematical terms above and below an arrow

LaTeX input	Output

`\begin{equation*}`
`  \mathrm{H_2 + D_2} \underset{\mathrm{Ni}}`
`    {{}^{\underrightarrow{~\mathrm{600\text{--}750\,^oC}~}}}`
`      \mathrm{2HD}`
`\end{equation*}`
	$\mathrm{H_2 + D_2} \underset{\mathrm{Ni}}{{}^{\underrightarrow{~\mathrm{600\text{--}750\,^oC}~}}} \mathrm{2HD}$

`\begin{equation*}`
`  \mathrm{H_2 + D_2} \overset{\mathrm{Ni}}`
`    {{}_{\overrightarrow{~\mathrm{600\text{--}750\,^oC}~}}}`
`      \mathrm{2HD}`
`\end{equation*}`
	$\mathrm{H_2 + D_2} \overset{\mathrm{Ni}}{{}_{\overrightarrow{~\mathrm{600\text{--}750\,^oC}~}}} \mathrm{2HD}$

e.g., ‘`$\underset{i=1}{\overset{n}{\sum}}$`’ or ‘`$\overset{n}{\underset{i=1}{\sum}}$`’ in this line produces $\underset{i=1}{\overset{n}{\sum}}$.

For stacking multiple lines above or below of a symbol, the `\substack{}` command or the `subarray` environment may be used. The lines are center aligned under `\substack{}`, while their alignment can be controlled in the `subarray` environment. Some examples of these options are shown in Table 12.16, where the lines under the `subarray` environment are left aligned through the option `l` (other option is `c` for center alignment).

Table 12.16 Stacking multiple mathematical lines above or below of a symbol

LaTeX input	Output

`\begin{equation*}`
`  \sum_{\substack{i=1\\ i\in\Omega_{\text{old}}}}`
`\end{equation*}`
	$\sum_{\substack{i=1\\ i\in\Omega_{\text{old}}}}$

`\begin{equation*}`
`  \prod_{i=1}^{\substack{i=n\\ n=\text{Sl.No}}}`
`\end{equation*}`
	$\prod_{i=1}^{\substack{i=n\\ n=\text{Sl.No}}}$

`\begin{equation*}`
`  \sum_{\begin{subarray}{l} i=1\\ i\in\Omega_{\text{old}}`
`        \end{subarray}}`
`\end{equation*}`
	$\sum_{\begin{subarray}{l} i=1\\ i\in\Omega_{\text{old}} \end{subarray}}$

`\begin{equation*}`
`  \prod_{i=1}^{\begin{subarray}{l} i=n\\ n=\text{Sl.No}`
`        \end{subarray}}`
`\end{equation*}`
	$\prod_{i=1}^{\begin{subarray}{l} i=n\\ n=\text{Sl.No} \end{subarray}}$

There is another slightly different command, `\sideset{aleft}{aright}`, which prints `aleft` and `aright`, respectively, on the left and right sides of a symbol, like $\sum$ or $\prod$. Provision is also there for printing four different terms, as superscripts and subscripts, on the four corners of a symbol. For example, `$\sideset{a}{b}\prod$` and `$\sideset{^1_2}{^3_4}\prod$` will print $\sideset{a}{b}\prod$ and $\sideset{^1_2}{^3_4}\prod$ respectively.

12.8 Side-by-Side Equations*

If required for some purpose (say, for comparison), sets of equations can be produced side-by-side along the width of a page. Generally, the **gathered**, **aligned** and **alignedat** environments are used in such cases, nesting in a math-mode, such as the **equation** or **displaymath** environment, or **\[\]** mode. Vertical alignment of the sets of equations can also be made through an optional argument to the environments, whose permissible values are **c** for center alignment, **b** for bottom alignment and **t** for top alignment. These environments are very similar, respectively, with the **gather**, **align** and **alignat** environments discussed in §11.5 on page 107. Two applications of the pair of **aligned** and **gathered** environments are shown in Table 12.17. In the first application, both the sets of equations are center aligned (by default), while in the second application, the first set is bottom aligned and the second set is top aligned.

Table 12.17 Side-by-side equations along the page width

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td>

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

</td><td>

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

</td></tr>
<tr><td>

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

</td><td>

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

</td></tr>
</table>

Hour 13

User-Defined Macros

LaTeX provides many in-built commands and environments for preparing a document. Besides those, it permits to define new commands and environments. Moreover, in-built ones can also be redefined to alter their behaviors. Many such cases are already addressed in previous Hours, which are systematically discussed in this Hour. In this book, although the original LaTeX syntax are printed in **red colored** (for online version) **and boldfaced sans serif fonts** for their clear distinction, all user-defined syntax are printed in normal black color so that a reader is not misled.

13.1 Defining New Commands

It is a tedious job to insert a long command or a combination of commands, or even a piece of texts, if it is to be used repeatedly in a document. In that case, some short commands can be defined to represent such long items conveniently[1]. A new command is defined in the preamble through the **\newcommand{**`newc`**}{**`aval`**}** or **\providecommand{**`newc`**}{**`aval`**}** command, where `newc` is the new command to be defined and `aval` is the attribute to be represented by `newc`. In general, the name of a new command should be alphabetic only, not to start with '`end`' and not to match with an existing command. In the case of **\newcommand{}{}**, an error message will be generated if a new command matches with an existing command, while **\providecommand{}{}** will retain the existing command without any message, i.e., whether the new command is defined or the existing one is retained (therefore, **\providecommand{}{}** will not be discussed any more in this book).

[1] A new short command can defined to represent conveniently a long command or a combination of commands or even a piece of texts, if it is to be used repeatedly in a document.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_13

13.1.1 New Commands Without Argument

It is discussed in §1.5.1 on page 5 as well as seen in previous Hours that some commands work on their own, while others ask a user to provide some inputs. Table 13.1

Table 13.1 Defining new commands without argument

Definition of new command (in the preamble)	Meaning
`\newcommand{\bs}{$\backslash$}`	'`\bs`' to print '\'
`\newcommand{\xv}{\mbox{\boldmath$x$}}`	'`\xv`' to print '$\boldsymbol{x}$'
`\newcommand{\veps}{\ensuremath{\varepsilon}}`	'`\veps`' to print 'ε'
`\newcommand{\cg}{\it Center of Gravity\/}`	'`\cg`' to print '*Center of Gravity*'

shows how a new command without any argument (i.e., without any input from a user) can be defined in the preamble and its meaning when used in the body of a document. The `\bs` command is defined to represent the math-mode `\backslash` command. However, it is applicable in text-mode only. If used in math-mode, the first `$` sign in its definition will quit the math-mode. To alleviate such errors, the inline math-mode may be generated through `\mbox{}`, which works in both text-mode and math-mode. Another option is to use `\ensuremath{}`, which always processes its argument in math-mode regardless the mode in which it is used. The applications of `\mbox{}` and `\ensuremath{}` are shown in Table 13.1 in the definitions of the `\xv` and `\veps` commands, which also show the representation of a combination of some existing commands by a single new command. On the other hand, the `\cg` command in Table 13.1 shows that, not only existing commands but a piece of normal texts can also be represented by a new command.

Note that, if ended by an alphabet, a user-defined new command (like `\bs`, `\xv`, `\veps` and `\cg` in Table 13.1) is also to be terminated by `\␣` in order to protect the trailing blank space. Without trailing `\␣`, the texts next to a command would be printed in continuation of the command, e.g., '`\xv␣is a vector`' will print '$\boldsymbol{x}$is a vector' (without any gap between '$\boldsymbol{x}$' and 'is'), while '`\xv\␣is a vector`' will print '$\boldsymbol{x}$ is a vector'.

13.1.2 New Commands with Mandatory Arguments

In the case of presenting a particular scenario repeatedly, a general command would be a more preferred one, which can be applied to different cases by changing the requirements of a user. For example, the `\xv` command of Table 13.1 is fixed to print `x` as a vector. If `y` is also to be presented as a vector in some cases, it will require to define another similar command, say `\yv`. This will not only increase the size of the preamble, but it would be difficult also to remember the commands defined for different cases. Therefore, instead of case-based fixed commands like `\xv` and `\yv`, it would be convenient to define a single command with some arguments to take the requirements of a user. Such a new command is defined through `\newcommand{}[]{}` as `\newcommand{newc}[n]{..{#1}..{#2}..{#n}..}`, where `newc` is the new command to be defined, `n` is the number of arguments (a command can have a maximum of 9 arguments), and `#1`, `#2`, …, `#n` are the serial numbers of the arguments. Although the arguments need not appear in order in the definition of a new command, they must be supplied in order during application, each argument in separate `{}`.

A number of examples of defining new mathematical commands having arguments are shown in Table 13.2. The definitions of the new commands may be

Table 13.2 Definition and application of new commands with mandatory arguments

Definition of new command (in the preamble)	Meaning
`\newcommand{\vctr}[1]{\mbox{\boldmath{$#1$}}}`	`#1` as a vector.
`\newcommand{\pde}[2]{\ensuremath{%`	
`\frac{\partial #2}{\partial #1}}}`	Partial derivative of `#2` w.r.t. `#1`.
`\newcommand{\ode}[2]{\ensuremath{\frac{d#2}{d#1}}}`	Ordinary derivative of `#2` w.r.t. `#1`.
`\newcommand{\oded}[2]{\ensuremath{%`	
`\frac{d^2#2}{d#1^2}}}`	Second order OD of `#2` w.r.t. `#1`.
`\newcommand{\odp}[2]{\ensuremath{\frac{d}{d#1}(#2)}}`	OD of `(#2)` w.r.t. `#1`.
`\newcommand{\intg}[2]{\ensuremath{\int(#2)\,d#1}}`	Integration of `(#2)` w.r.t. `#1`.
`\newcommand{\dint}[4]{\ensuremath{%`	Integration of `(#2)` w.r.t. `#1`
`\int_{#3}^{#4}(#2)\,d#1}}`	from `#3` to `#4`.
`\newcommand{\lmt}[4]{\ensuremath{%`	
`\lim_{#3\to #4}\frac{#1}{#2}}}`	Limit of `#1/#2` for `#3`→`#4`.

SN	LaTeX input	Output
1	`\vctr{x}.\vctr{y} = $z$`	$\boldsymbol{x}.\boldsymbol{y} = z$
2	`\pde{y}{x}`	$\frac{\partial x}{\partial y}$
3	`\ode{y}{x}`	$\frac{dx}{dy}$
4	`\oded{y}{x}`	$\frac{d^2x}{dy^2}$
5	`\odp{y}{x^2+3xy-5}`	$\frac{d}{dy}(x^2+3xy-5)$
6	`\intg{x}{x^5+4x^2-10}`	$\int(x^5+4x^2-10)\,dx$
7	`\dint{p}{p^3q+5pq-q}{0}{3}`	$\int_0^3(p^3q+5pq-q)\,dp$
8	`\dint{p}{p^3q+5pq-q}{v}{}`	$\int_v(p^3q+5pq-q)\,dp$
9	`\lmt{x^2+3x-10}{x-2}{x}{2}`	$\lim_{x\to 2}\frac{x^2+3x-10}{x-2}$

difficult to understand, which is cleared through their applications. The `\,` command is used between `(#2)` and `d#1` in the definitions of `\intg` and `\dint` for maintaining a small gap before a differential, which may be viewed before dx or dp in applications 6–8. It is also to be noted that if there is nothing to provide against a mandatory argument of a command, it can be left just by empty `{}`, which is demonstrated in application 8 by keeping the upper limit of `\dint` empty. In this way, `\dint` can be used for indefinite integral also, by leaving its last two arguments empty.

Tables 13.1 and 13.2 show the process of defining a new command to represent an existing command or a combination of existing commands, or even a piece of normal texts. The `\newcommand{}{}` can also be used to reduce the number of arguments of an existing command. For example, if texts in red color are to be produced repeatedly, as done in this book (for online version), a shorter command, say `\tred`, may be defined to replace the repeated use of the long `\textcolor{red}`. In that case, instead of `\textcolor{red}{atext}`, just `\tred{atext}` can be used for printing `atext` in red color.

13.1.3 New Commands with Optional Arguments

The arguments of all the new commands of Table 13.2 in §13.1.2 are mandatory. An optional argument can also be assigned to a user-defined new command[2]. Consider the case of `\xv` defined in Table 13.1, which prints `x` as a vector. The same `\xv` command with an optional argument can be used to print another letter also as a vector, instead of using `\vctr{}` shown in Table 13.2. A new command with an optional argument is defined in a similar way as shown in Table 13.2, but with an additional optional argument to `\newcommand{}[]{}`, i.e., through `\newcommand{}[][]{}` as `\newcommand{newc}[n][farg]{..{#1}..{#2}..{#n}..}`, where `farg` is the default first argument (which is optional) of the new command `newc`. Table 13.3 shows the

Table 13.3 Definition and application of new commands with optional arguments

Definition of new command (in the preamble)	Meaning
`\newcommand{\xv}[1][x]{\mbox{\boldmath{$#1$}}}`	Optional **`#1`** or default `x` as a vector.
`\newcommand{\drv}[2][y]{\ensuremath{\frac{d}{d#1}(#2)}}`	Ordinary derivative of **`(#2)`** w.r.t optional **`#1`** or default `y`.

SN	LaTeX input	Output
1	`\xv\␣is a vector.`	$\boldsymbol{x}$ is a vector.
2	`\xv[y] is also a vector.`	$\boldsymbol{y}$ is also a vector.
3	`\drv{x}`	$\frac{d}{dy}(x)$
4	`\drv[x]{\sin x}`	$\frac{d}{dx}(\sin x)$

definitions and applications of two new commands, each with an optional argument. The `\xv` command has only one argument, which is optional. By default `\xv` prints $\boldsymbol{x}$, otherwise its optional argument if provided, say `\xv[P]` will print $\boldsymbol{P}$ (an optional argument is provided in `[]`). On the other hand, the `\drv` command has two arguments – the first one is optional and the second one is mandatory. In the absence of the optional argument, `\drv` differentiates its mandatory argument with respect to default `y`, otherwise with respect to its optional argument if provided.

13.2 Redefining Existing Commands*

In some applications, the default style of an existing command may need to be altered, which is done by redefining the command[3]. An existing command is redefined in the preamble through the `\renewcommand{rcom}[n]{astyle}` command, where `rcom` is the existing command to be redefined, and `astyle` is its opted new style in terms of `n` number of arguments. For example, `\renewcommand{\labelitemi}{{\small$\vartriangleright$}}` replaces the bullet marking of

---

[2]Only one optional argument is permitted to a user-defined new command.

[3]An existing command can be redefined to alter its default style.

the items with '▷' under the first level of the **itemize** environment. Some heading commands generate their label-words, like **\chapter{}** generates '`Chapter`' before its serial number. Such default label-words are listed in Table 13.4, which can be altered

Table 13.4 Heading commands and their default label-words

Heading Command	Default label-word	Heading Command	Default label-word
\abstractname	Abstract	\indexname	Index
\appendixname	Appendix	\listfigurename	List of Figures
\bibname	Bibliography	\listtablename	List of Tables
\chaptername	Chapter	\partname	Part
\contentsname	Contents		

through **\renewcommand{}[]{}**, e.g., **\renewcommand{\chaptername}{**`Unit`**}** for replacing '`Chapter`' by '`Unit`', or **\renewcommand{\abstractname}{**`Summary`**}** for replacing '`Abstract`' by '`Summary`'. Many more examples of redefining existing commands have already been provided in previous Hours, specially in Hours 4–6, 8, and 9.

Note that the first argument, if any, of a redefined command also can be made optional in a similar way as that of a new command stated in §13.1.3, i.e., through **\renewcommand{**`rcom`**}[**`n`**][**`farg`**]{**`astyle`**}** with `farg` as the default first argument of `rcom`.

Also note that the command to be redefined cannot be repeated in the second argument of **\renewcommand{}{}**, e.g., **\renewcommand{\alpha}{**`Symbol-`**\alpha}** is not permitted. In such a requirement, a command may be redefined in one of the following two processes:

1. First, save the existing command by another name, and then redefine the former in terms of the latter. As an example, first saving **\textcolor** as `\oldtextcolor` using '**\let**`\oldtextcolor`**\textcolor**', **\textcolor** then can be redefined as **\renewcommand{\textcolor}**`[2][red]`**{**\oldtextcolor**{#1}{#2}}** to print its argument in default red color or in the supplied optional color, i.e., **\textcolor{**`atext`**}** to print `atext` in red color and **\textcolor[**`acol`**]{**`atext`**}** to print `atext` in `acol` color.
2. First, obtain the LaTeX's internal coding of the existing command, and then redefine the command in terms of that coding. For example, **\sigma** can be redefined as **\renewcommand{\sigma}{\mbox{\boldmath{$\mathchar"11B$}}}** in terms of its internal coding '**\mathchar"11B**' so as to print $\boldsymbol{\sigma}$ by **\sigma**. LaTeX's internal coding of a command can be obtained by using **\show**, e.g., the use of **\show\sigma** somewhere in the LaTeX input file will pause its compilation displaying '`> \sigma=\mathchar"11B`', which means that the internal coding of **\sigma** is **\mathchar"11B**.

13.3 Defining New Environments

Like new commands discussed in §13.1 on page 125, user-specific new environments can also be defined. A new environment is usually defined in the preamble in terms of an existing environment for obtaining its content in a slightly modified pattern. Moreover, a completely new pattern can also be obtained as per requirement.

13.3.1 New Environments Without Argument

A new environment can be defined as **\newenvironment{**nenv**}{**cstart**}{**cend**}**, where `nenv` is the new environment, and `cstart` and `cend` are, respectively, the commands for starting and ending the new environment. Table 13.5 shows the

Table 13.5 Definition and application of new environments without argument

LaTeX input	Output

<pre>\documentclass{article}
\newenvironment{itemem}%
                {\begin{itemize}\em}{\end{itemize}}
%
\newenvironment{boxednote}{\begin{center}\em%
    \begin{tabular}{|p{0.8\textwidth}|}\hline}%
        {\\ \hline\end{tabular}\end{center}}
%
\begin{document}
    \begin{itemem}
    \item Emphasized items.
    \item Modified itemize environment.
    \end{itemem}
%
    \begin{boxednote}
    This is a new environment for ...
    \end{boxednote}
\end{document}</pre>
	• *Emphasized items.*
	• *Modified itemize environment.*
	
	This is a new environment for producing important notes and observations in emphasized fonts inside a box.

definitions and applications of two new environments. The first new environment is `itemem`, which is defined by modifying the existing **itemize** environment for producing its contents in emphasized fonts. It is done through **\em** inserted in the second argument of **\newenvironment{}{}{}**. Not only **\em**, other commands including a new one or a redefined one can also be added for obtaining a desired pattern. For example, the bullet marks (•) in Table 13.5 can be replaced by check-mark (✓) by defining the environment as '**\newenvironment{**itemem**}** **{\begin{itemize}\renewcommand{\labelitemi}{\checkmark}}{\end{itemize}}**', where **\labelitemi** is redefined as **\checkmark** (refer §6.1.2 on page 53 for detail).

The second new environment in Table 13.5 is `boxednote`, defined by combining two existing environments for producing emphasized notes in a box. The new environment employs the **tabular** environment for producing a single-column table as a box, which is horizontally center aligned through the **center** environment. Finally, the **\em** command is used for emphasizing the contents of the new environment.

13.3.2 New Environments with Arguments

Similar to new commands discussed in §13.1.2 on page 126, a new environment can also be provided with some mandatory arguments, in which case it is to be defined through **\newenvironment{**`nenv`**}[**`n`**]{**cstart**}{**cend**}**, where `n` is the number of arguments to be provided by a user (a maximum of nine arguments can be provided). For example, in order to put a boldface title to the contents of the `boxednote` environment, the first line of its definition in Table 13.5 can be replaced with '**\newenvironment{**`boxednote`**}[**`1`**]{\begin{center}{\bf #**`1`**} \em**', and then the environment can be started in the **document** environment as **\begin{**`boxednote`**}{**`atitle`**\\}**, where `atitle` is the title of the contents to be printed in boldface fonts in a new line (it will be printed in a new line because of the line break command ****).

Further, an optional argument is also permitted to a user-defined environment. As in the case of new commands stated in §13.1.3 on page 128, a new environment with an optional argument is defined with an additional optional argument to **\newenvironment{}[]{}{}**, i.e., through **\newenvironment{}[][]{}{}** as **\newenvironment{**`nenv`**}[**`n`**][**`farg`**]{**`cstart`**}{**`cend`**}**, where `farg` is the default first argument (which is optional) of the new environment `nenv`. As an example, consider again the above case of the `boxednote` environment. In order to put an optional title to the contents of `boxednote`, the first line of its definition in Table 13.5 can be replaced with '**\newenvironment{**`boxednote`**}[**`1`**][]{\begin{center}{\bf #**`1`**}\em**'. If the environment is now started simply as **\begin{**`boxednote`**}**, its contents will be printed without any title because of the empty second optional argument in its definition. However, if it is started as **\begin{**`boxednote`**}[**`atitle`**\\]**, the optional argument `atitle` will be printed in boldface fonts in a new line as the title of the contents.

13.3.3 Theorem-Like Environments

Many mathematical documents include theorems and other theorem-like structures, such as definitions, lemmas, and propositions. A non-mathematical document may also contain similar structures, like principles, laws, assumptions, etc. For defining any such environment, LaTeX provides the **\newtheorem{**`akey`**}{**`nenv`**}[**`aunit`**]** command, where `nenv` is the name of the new environment and `akey` is its keyword, while optional `aunit` is the name of the sectional unit (like **chapter** or **section**) based on which the environment is to be numbered. The definitions of some of such environments are shown in Table 13.6 on the next page with or without an optional argument. Each theorem-like environment is numbered, preceded by the name of the environment. Numbering is done according to the optional argument provided in the

Table 13.6 Defining theorem-like environments

Definition	**Meaning**
\newtheorem{thm}{Theorem}[chapter]	'thm' for writing chapter-wise 'Theorem'
\newtheorem{dfn}{Definition}[chapter]	'dfn' for writing chapter-wise 'Definition'
\newtheorem{cor}{Corollary}[section]	'cor' for writing section-wise 'Corollary'
\newtheorem{lem}{Lemma}[section]	'lem' for writing section-wise 'Lemma'
\newtheorem{prop}{Proposition}	'prop' for writing 'Proposition'
\newtheorem{prf}{Proof}	'prf' for writing 'Proof'
\newtheorem{princ}{Principle}	'princ' for writing 'Principle'
\newtheorem{rul}{Rule}	'rul' for writing 'Rule'

Table 13.7 Application of the user-defined 'Definition' environment

LaTeX input	**Output**
\begin{dfn}[\bf Center of Mass]\label{dfn-cm} This is the point at which the entire mass of a body of uniform density can be assumed to be concentrated. \end{dfn} % \begin{dfn}{\bf Center of Gravity:} This is the point though which the resultant of the gravitational forces of all elemental weights of a body acts.\label{dfn-cg} \end{dfn} % Definition~\ref{dfn-cm} defines center of mass, while Definition~\ref{dfn-cg} …	**Definition 13.1 (Center of Mass)** *This is the point at which the entire mass of a body of uniform density can be assumed to be concentrated.* **Definition 13.2 Center of Gravity:** *This is the point though which the resultant of the gravitational forces of all elemental weights of a body acts.* Definition 13.1 defines center of mass, while Definition 13.2 defines center of gravity.

definition of the environment, otherwise globally if no option is provided. Table 13.7 shows an example of the Definition environment, which is defined in Table 13.6 with the keyword dfn and optional argument **chapter**.

Like any other numbered item, a theorem-like environment can also be labeled and referred through a keyword. An environment can be labeled anywhere within its body. This is shown in Table 13.7, where the first dfn environment is labeled at the very beginning of its body and the second one is labeled at the bottom of its body.

On the other hand, a theorem-like environment can be provided a title also. It is to be provided after **\begin{}** as an optional argument in **[]** or explicitly in **{}**. Both the styles are shown in the two dfn environments in Table 13.7. Notice that, in the case of the optional argument, the title is printed in a pair of parentheses.

Further, it may be noticed in Table 13.7 that the contents of a theorem-like environment is printed in a new paragraph. Before defining such an environment through **\newtheorem{}{}[]** (refer Table 13.6), its style can be controlled using the **\theoremstyle{}** command defined in the **amsthm** package, e.g., **\theoremstyle{break}** for starting in a new line, or **\theoremstyle{plain}** for continuing in the same line. The **amsthm** package also defines the starred form of **\newtheorem{}{}**, i.e., **\newtheorem*{}{}** (without any optional argument), for producing unnumbered theorem-like environments. The **\newtheorem*{}{}** command may be useful for defining an environment which will be created one time only. It may also be useful if an environment is to be identified by a particular name rather than by a serial number, e.g., the definition of 'Center of Gravity' as shown in Table 13.7.

13.3.4 Floating Environments for Textual Materials*

Materials of some items, like a table or figure, are not desired to be split over pages. If the remaining blank space on the current page is not sufficient to fit such an item, LaTeX floats (shifts) it to a convenient place, such as the top or bottom of the current page or even on the next page. However, textual materials, like algorithms and computer programs (codings), are split over pages to avoid partially filled pages. Such materials can also be forced through a floating environment to appear together on a single page. For tables and figures, LaTeX provides two standard floating environments, namely **table** and **figure** respectively, which are discussed in Hours 7–10. Similar floating environments may be defined for textual materials also.

The **float** package provides the **\newfloat{}{}{}[]** command for defining a new floating environment in the preamble as **\newfloat{**`afloat`**}{**`vpos`**}{**`extn`**}[**`unit`**]**, where mandatory `afloat` is the name of the floating environment to be defined (e.g., `algorithm` or `program`), `vpos` is its vertical positioning (**h**, **b** or **t**, or a combination of them), `extn` is the extension of the auxiliary file for storing the captions of the floats (say, `flt`), and optional `unit` is to specify how the float will be numbered (e.g., **section** for section-wise numbering, or **chapter** for chapter-wise numbering). Some provisions are also available with floating environments, which are the following:

1. By default the first argument of **\newfloat{}{}{}[]** becomes the label-word for the caption of the defined float. It can be altered using **\floatname{**`afloat`**}{**`clabel`**}** (after **\newfloat{}{}{}[]**) for labeling the caption of the `afloat` environment by `clabel`.
2. The **\floatstyle{}** command can be used before **\newfloat{}{}{}[]** for defining the style of a float. The permissible styles (i.e., the argument of **\floatstyle{}**) are **plain**, **boxed** and **ruled**. The **plain** style prints the caption below the float regardless the position of the **\caption{}** command in the floating environment. The **boxed**-style produces the float in a box and prints the caption below the box. On the other hand, the **ruled**-style prints the caption above the float and also puts it between two horizontal lines. Moreover, another horizontal line is produced below the float.
3. By default a maximum of three floats are permitted on a single page. The number can be changed by redefining the counter **totalnumber** through the **\setcounter{}{}** command, e.g., **\setcounter{totalnumber}{**`10`**}** for allowing a maximum of 10 floats on a single page, if the available space of a page is sufficient for that. This counter is equally applicable to the standard **table** and **figure** floating environments.
4. Analogous to the **\listoftables** and **\listoffigures** commands used for producing lists of tables and figures respectively (§16.1 on page 153 discusses in detail), the **\listof{}{}** command can be used for producing the list of floats of a given type, e.g., **\listof{**`program`**}{**`List of Computer Programs`**}** will produce a list of `program`-type floats under the heading ‘`List of Computer Programs`’.

Table 13.8 on the next page shows the process for defining and applying two new user-defined floating environments, namely `algorithm` and `program`, for presenting algorithms and computer programs, respectively. Through the **\floatstyle{}**, **\newfloat{}{}{}[]**, and **\floatname{}{}** commands (in order), the **ruled**-style `algorithm`

Table 13.8 Floats for presenting algorithms and computer programs

LaTeX input

```
\documentclass[11pt,a4paper]{article}
\usepackage{float}
\floatstyle{ruled}
\newfloat{algorithm}{hbt}{alg}[section]
\floatname{algorithm}{Algorithm}
\floatstyle{boxed}
\newfloat{program}{hbt}{prg}[section]
\floatname{program}{Program}
%
\begin{document}
...
\section{Finding the maximum}
The main steps ... Algorithm~\ref{algo:max}.
\begin{algorithm}
\caption{Maximum of $n$ data points.}
\label{algo:max}
\begin{enumerate}
\item Read the number of data points $n$.
\item Read the data point $a_i$;
      $i=1$ to $n$.
\item Set {\it max} $=a_1$.
.
:
\end{enumerate}
\end{algorithm}
%
Algorithm~\ref{algo:max} is coded in the
{\tt C} computer ... Program~\ref{prog:max}.
\begin{program}
\caption{Maximum of $n$ data points.}
\label{prog:max}
\begin{verbatim}
#include<stdio.h>
#include<math.h>
int main()
{
    int n, a[101];
    int i, max;

    printf("Number of points = ");
    :
    return(0);
}
\end{verbatim}
\end{program}
...
\end{document}
```

Output

5 Finding the maximum

The main steps for finding and printing the maximum from a given data set of n points are shown in Algorithm 5.1.

Algorithm 5.1 Maximum of n data points.

1. Read the number of data points n.
2. Read the data point a_i; $i = 1$ to n.
3. Set $max = a_1$.
4. If $max < a_i$, set $max = a_i$; $i = 2$ to n.
5. Print *max* as the maximum of given n number of data points.

Algorithm 5.1 is coded in the C computer programming language, which is shown here in Program 5.1.

```
#include<stdio.h>
#include<math.h>
int main()
{
    int n, a[101];
    int i, max;

    printf("Number of points = ");
    scanf("%d", &n);
    for(i = 1; i <= n; i++)
    {
        printf("a[%d] = ", i);
        scanf("%d", &a[i]);
    }

    max = a[1];
    for(i = 2; i <= n; i++)
        if(max < a[i]) max = a[i];

    printf("\nLargest value = ");
    printf("%d\n", max);

    return(0);
}
```

Program 5.1: Maximum of n data points.

environment is defined with labeling its captions by `Algorithm` and storing them in an auxiliary file of `alg` extension. Similarly, the **boxed**-style `program` environment is defined with labeling its captions by `Program` and storing them in an auxiliary file of `prg` extension. Both the environments are opted to be numbered section-wise, and positioned here (**h**), bottom of the page (**b**) or top of the next page (**t**) as per the situation at hand. Then the applications of the environments are shown in the body of the LaTeX input file. The computer coding in the `program` environment is inserted through the **verbatim** environment for printing the contents, as inserted, ignoring the LaTeX mode (§18.5 on page 173 discusses verbatim texts in detail). Note that, as shown in Table 13.8, a float environment can also be labeled and referred like any other environment.

13.4 Redefining Existing Environments*

It is stated in §13.3 on page 130 that a new environment can be defined in terms of an existing environment for obtaining its contents in a slightly modified pattern. Instead of creating a new environment, an existing environment can also be redefined for modifying its effect. However, redefinition of an existing environment will change its effect in the entire document. Hence, instead of redefining an existing environment, a new one should be defined, if changes are required only in certain portions of a document.

Similar to defining a new environment, an existing environment is redefined through the `\renewenvironment{nenv}{cstart}{cend}` command, where `nenv` is the name of the existing environment which is to be redefined, and `cstart` and `cend` are, respectively, the new commands for starting and ending the environment. Consider an example that a nested list of unnumbered items is to be prepared by eliminating inter-item spacing as well as emphasizing the items of every alternate loop. This can be done by redefining the **itemize** environment as shown in Table 13.9 by reproducing

Table 13.9 Redefining the unnumbered listing environment **itemize**

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>
\documentclass[11pt, a4paper]{article}
\usepackage{paralist}
\renewenvironment{itemize}%
        {\em\begin{compactitem}}{\end{compactitem}}
%
\begin{document}
\begin{itemize}
\item India
     \begin{itemize}
     \item Assam
          \begin{itemize}
          \item Sonitpur
               \begin{itemize}
                    \item Tezpur
                    \item Dhekiajuli
                    \item Balipara
               \end{itemize}
          \item Kamrup
          \item Cachar
          \end{itemize}
     \item Bihar
     \item Punjab
     \end{itemize}
\item Pakistan
\item Sri Lanka
\end{itemize}
\end{document}
</pre></td><td>

- *India*
  - Assam
    - *Sonitpur*
      - Tezpur
      - Dhekiajuli
      - Balipara
    - *Kamrup*
    - *Cachar*
  - Bihar
  - Punjab
- *Pakistan*
- *Sri Lanka*

</td></tr>
</table>

the list given in Table 6.7 on page 54. In this example, the **itemize** environment is redefined as the **compactitem** environment, which is defined in the **paralist** package. Further, the **\em** command is used in the redefinition of **itemize** for obtaining emphasized texts.

Hour 14

Bibliography with LaTeX

The provision for labeling and referring various numbered materials, like sectional units, environments, foot notes, enumerated items, page numbers, etc., is discussed in the previous relevant Hours. Apart from that, LaTeX can generate very elegant bibliographic references on its own. A list of bibliographic references is generated either directly entering the detail of the references (i.e., the publications or documents to be cited) in the LaTeX input file, or combining LaTeX with its companion program BIBTEX. The former provision is discussed in this Hour (the latter is discussed in Hour 15 on page 141).

14.1 Preparation of Bibliographic Reference Database

The bibliographic reference list with LaTeX is generated through the **thebibliography** environment (generally at the end, just before **\end{document}**). The entry of a reference under the **thebibliography** environment consists of two mandatory parts – (1) a user-defined unique citation key, which can be used for citing the reference, and (2) detail of the reference like author, title, journal, volume, pages, etc.

As shown in Table 14.1 on the next page, the citation key of a reference, along with an optional identifier, is inserted through the **\bibitem[]{}** command as **\bibitem[**`ident`**]{**`ckey`**}**, where `ident` is the identifier and `ckey` is the citation key of the reference (without the optional argument of **\bibitem[]{}**, i.e., in the **\bibitem{}** form, references are listed by their serial numbers in Arabic numerals in square brackets). The detail of the reference may be inserted as plain texts or each piece of information through a separate **\newblock** command. Since the **thebibliography** environment processes the details of a reference in simple text-mode without any formatting provision, it is to be instructed explicitly by a user, e.g., '**\newblock {\em** `Introduction to Optimum Design`**}**' for producing '*Introduction to Optimum Design*', or '**\newblock** `The` **ϵ**`-constraint approach`' for producing 'The ϵ-constraint approach'.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_14

The **thebibliography** environment takes a mandatory argument to specify the number of starting spaces (columns) to be reserved for printing the serial numbers or identifiers of the references. For example, 4 number of `0`'s are used in **\begin{thebibliography}{**`0000`**}** in Table 14.1 for reserving the first four columns for

Table 14.1 Bibliographic references through the **thebibliography** environment

```
\documentclass{article}
:
\begin{document}
:
\begin{thebibliography}{0000}
\bibitem[1989]{Arora-1989}
Arora J. S. {\em Introduction to Optimum Design}. McGraw-Hill, 1989.
%
\bibitem[2001]{Deb-2001}
\newblock Deb K.
\newblock {\em Multi-Objective Optimization using Evolutionary Algorithms}.
\newblock John Wiley \& Sons, 2001.
\end{thebibliography}
%
\end{document}
```

that purpose (other digits or letters can also be used for reserving columns). In the case of identifiers, the longest identifier of all the references is usually used as the argument of **\begin{thebibliography}{ }**. Similarly, the serial number of the last reference may be used as the argument of **\begin{thebibliography}{ }** in the numbering system.

If the same references are to be included in different documents, those can be stored in a separate file as the database, instead of coding them in every document. In that case, just the name of the reference database file is to be included in a document, using the **\input{ }** command, for reading the references stored in it. Such a sample reference database file prepared in the **thebibliography** environment, named as `mybib.bib` (a reference database file is written with `bib` extension), is shown in the left column of Table 14.2 on the following page and its inclusion in a LATEX input file is shown in the right column. Note that the entry under a **\newblock** command can be split into multiple lines (all texts prior to a **\newblock** command are considered under the previous **\newblock** command).

14.2 Citing Bibliographic References

In the contents of a LATEX input file, a reference is cited through the **\cite{**`ckey`**}** command, where `ckey` is the unique citation key of the reference (the **\bibitem[ ]{ }** and **\cite{ }** commands work much like the **\label{ }** and **\ref{ }** commands discussed in earlier Hours).

Table 14.2 Bibliographic reference database compatible to the **thebibliography** environment

`.bib` **database file**	**Inclusion of** `.bib` **database file**

`% mybib.bib`
`\begin{thebibliography}{00}`
`\bibitem{Beven-2000}`
`\newblock Beven, K.`
`\newblock {\em Rainfall-Runoff Modelling,`
`          The Primer.}`
`\newblock John Wiley \& Sons, Chichester; 2000.`
`%`
`\bibitem{Black-2004}`
`\newblock Black, P. E.`
`\newblock {\em Hamming Distance.}`
`\newblock www.nist.gov/dads/HTML/hammingdist.html;`
`           December, 2004.`
`%`
`\bibitem{Schaerf-1999}`
`\newblock Schaerf, A.`
`\newblock A survey of automated timetabling.`
`\newblock {\em Artificial Intelligence Review},`
`          1999, 13:87-127.`
`\end{thebibliography}`
	`\documentclass{article}`
	⋮
	`\begin{document}`
	⋮
	`\input{mybib.bib}`
	`\end{document}`

As shown in Table 14.3, references are listed in the output under the heading ‘`References`’ (the heading in the document-class of **article** is ‘`References`’, while it is ‘`Bibliography`’ in the document-class of **book** and **report**). Each reference is marked either by a serial number in Arabic numeral or by its optional identifier, if any, e.g., `[1]` and `[2001]` in Table 14.3. Accordingly, the references are cited in the output either by their serial numbers or identifiers (both the options are shown in the same document of Table 14.3 for illustrative purpose only).

Table 14.3 Citing bibliographic references of the **thebibliography** environment

LaTeX input	**Output**

`Arora~\cite{Arora-1989} may be referred for`
`classical optimization, and`
`Deb~\cite{Deb-2001} for multi-objective`
`evolutionary algorithms.`
`%`
`\begin{thebibliography}{0000}`
`\bibitem{Arora-1989}`
`Arora J. S.`
`{\em Introduction to Optimum Design.}`
`McGraw-Hill, 1989.`
`%`
`\bibitem[2001]{Deb-2001}`
`\newblock Deb K.`
`\newblock Multi-Objective Optimization`
`using Evolutionary Algorithms.`
`\newblock John Wiley \& Sons, 2001.`
`\end{thebibliography}`
	Arora [1] may be referred for classical optimization, and Deb [2001] for multi-objective evolutionary algorithms.
	
	References
	
	[1] Arora J. S. *Introduction to Optimum Design*. McGraw-Hill, 1989.
	
	[2001] Deb K. Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons, 2001.

A reference can be cited with an optional note also, for which the **\cite[ ]{ }** command is to be used as **\cite[**`anote`**]{**`ckey`**}**, where `anote` is the optional note and `ckey` is the mandatory citation key. For example, **\cite[**`pages 43--47`**]{**`Datta15`**}** will produce, say `[25, pages 43–47]`.

In many applications, citation markings may need to be superscribed (i.e., to be put as superscripts), instead of putting in the same line with texts. For this, simply the **overcite** package may be loaded in the preamble, which will automatically produce superscribed citations without requiring any more change[1]. However, no note can be put with a superscribed citation, i.e., **\cite[ ]{ }** command will not work properly in the presence of the **overcite** package.

14.3 Compiling thebibliography Based LaTeX Input File

If the bibliography is generated through the **thebibliography** environment, the compilation of the LaTeX input file is to be changed from that addressed in §1.4 on page 4. In this case, it is to be compiled twice using the following two lines of commands:

```
$ latex myarticle
$ latex myarticle
```

where '`myarticle`' is the name of the LaTeX input file with '`tex`' extension. The second 'latex' command links the generated bibliographic references with LaTeX. The above two lines of commands will produce three files, namely `myarticle.aux`, `myarticle.log` and `myarticle.dvi` (§20.4.1 on page 199 discusses in further detail). As mentioned in §1.4, the '`myarticle.dvi`' file can be viewed in a document viewer or can be used to produce a '`.ps`' or a '`.pdf`' file.

---

[1] If the bibliographic reference list is produced through the **thebibliography** environment, superscribed citations of the references can be obtained just upon loading the **overcite** package.

Hour 15

Bibliography with the BIBTEX Program

The thebibliography environment discussed in Hour 14 cannot differentiate the types of references, i.e., whether an article or a book. Moreover, even if not cited, all the references inserted in the environment are printed in the output of a document. It has a drawback, particularly when references are to be included from a separate database file (refer Table 14.2 on page 139). A separate bibliographic database file is usually prepared so that it can be used in more than one document. However, all the references, stored in a database file, may not be required in a particular document. In that case, the environment fails to print only the selective references from such a database file.

The above drawbacks of the thebibliography environment can be overcome in the BIBTEX program, which prints a reference only if it is cited somewhere in the document. Moreover, the BIBTEX program follows certain structures for different types of references. It is stated in §14.1 that the thebibliography environment processes the entry of a reference without any formatting provision, which is to be set manually by a user. In fact, the BIBTEX program also internally prepares the list of references in the thebibliography environment only, but automatically follows some predefined structures according to the chosen bibliography style discussed in §15.2 on page 146.

15.1 Preparation of BIBTEX Compatible Reference Database

The entry of a reference in the BIBTEX program consists of three mandatory parts – (1) type of the reference, (2) a user-defined citation key which can be used for citing the reference, and (3) detail of the reference.

As shown in Table 15.1 on the next page, there are around 14 types of defined references, which are article (articles in journals or magazines), book (books), booklet (booklet type references), inbook (chapters or parts of books), incollection (parts of a book with separate titles), inproceedings (articles in conference proceedings), conference (articles in conference proceedings), manual (technical documentations), mastersthesis (Master degree theses), phdthesis (Ph.D theses),

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_15

Table 15.1 Types and fields of references under the BIBTEX program

	article	book	booklet	inbook	incollection	inproceedings, conference	manual	mastersthesis, phdthesis	misc	proceedings	techreport	unpublished
address	×	O	O	O	O	O	O	O	×	O	O	×
author	M	M_1	O	M_1	M	M	O	M	O	×	M	M
booktitle	×	×	×	×	M	M	×	×	×	×	×	×
chapter	×	×	×	M_2	O	×	×	×	×	×	×	×
edition	×	O	×	O	O	×	O	×	×	×	×	×
editor	×	M_1	×	M_1	O	O	×	×	×	O	×	×
howpublished	×	×	O	×	×	×	×	×	O	×	×	×
institution	×	×	×	×	×	×	×	×	×	×	M	×
journal	M	×	×	×	×	×	×	×	×	×	×	×
month	O	O	O	O	O	O	O	O	O	O	O	O
note	O	O	O	O	O	O	O	O	O	O	O	M
number	O	O_1	×	O_1	O_1	O_1	×	×	×	O_1	O	×
organization	×	×	×	×	×	O	O	×	×	O	×	×
pages	O	×	×	M_2	O	O	×	×	×	×	×	×
publisher	×	M	×	M	M	O	×	×	×	O	×	×
school	×	×	×	×	×	×	×	M	×	×	×	×
series	×	O	×	O	O	O	×	×	×	O	×	×
title	M	M	M	M	M	M	M	M	O	M	M	M
type	×	×	×	O	O	×	×	O	×	×	O	×
volume	O	O_1	×	O_1	O_1	O_1	×	×	×	O_1	×	×
year	M	M	O	M	M	M	O	M	O	M	M	O

M $\rightarrow$ mandatory field M_1 $\rightarrow$ one of them is mandatory M_2 $\rightarrow$ either one or both are mandatory
O $\rightarrow$ optional field O_1 $\rightarrow$ one of them (optional) × $\rightarrow$ not required

misc (uncommon references), **proceedings** (proceedings of an event), **techreport** (technical reports or working papers), and **unpublished** (unpublished references). The reference-type commands are preceded by **@** and they take a mandatory argument, e.g., **@article{**`ckey,rf1,rf2,...`**}**, where `ckey` is the citation key of a reference, and `rf1`, `rf2`, etc., are some mandatory and optional fields detailing the reference.

As shown in Table 15.1, a reference-type command takes some fields (out of around 21 commonly used reference fields), which are **address**, **author**, **booktitle**, **chapter**, **edition**, **editor**, **howpublished**, **institution**, **journal**, **month**, **note**, **number**, **organization**, **pages**, **publisher**, **school**, **series**, **title**, **type**, **volume**, and **year**. These fields can be entered in any order, which will be arranged automatically according to the chosen bibliography style as stated in §15.2 on page 146. There is no harm if an extra field is inserted or an acceptable field is left blank. A redundant field or a field with no data is automatically skipped by the BIBTEX program[1].

Data of all the fields of a reference-type command are processed in text-mode. Hence, accented and special characters are to be put in proper way,

[1]A redundant or empty field, as well as listing of non-cited documents, is automatically skipped by BIBTEX.

e.g., 'Jos\'e' for producing 'José' or 'Hungerl$\ddot{\mathrm{a}}$nder' for producing 'Hungerländer'. Data of a field can be inserted either in a pair of quotes or curly braces, e.g., **title="**A Practical Guide to **\LaTeX"** or **title={**A Practical Guide to **\LaTeX}**. Further processes for entering data in the argument of a reference-type command are explained below:

- *Citation key*: There is no format for a citation key, it can be a combination of any number of alphabets and numerals as well as some signs (like '+', '−', and ':') without any gap between two characters, e.g., Even-etal-1976 or Even+:1976.
- **address**: It could be the city or country of a publisher, venue of a conference, address of an institution or school, or URL of a webpage. A URL may be inserted through the **\url{}** command defined in the **url** package.
- **author**: The exact printing format of authors is decided by the chosen bibliography style as discussed in §15.2 on page 146 and §15.3 on page 147. In the reference database file, however, the names of all the authors may be inserted consistently or exactly as they appeared in the original individual documents to be referred. BIBTEX processes the name of an author in two parts only, the given name (first name) and the surname (family name). Hence, the following points may be noted:

  1. In the case of all words capitalized, the last word is treated as the surname and rest as the given name, e.g., in 'Krishna Prasad Rama Murthy', 'Murthy' will be treated as the surname and 'Krishna Prasad Rama' as the given name.
  2. In the presence of any noncapitalized word, however, all the remaining words starting from the first noncapitalized one are treated as the surname and rest as the first name, e.g., in 'Robert von der Smith', 'von der Smith' will be treated as the surname and 'Robert' as the given name. Similarly, the entire name 'von der Smith Robert' will be treated as the surname without a given name.
  3. If a surname contains multiple words, those may be inserted either in curly braces or at the starting with a comma, e.g., 'Pedro Jose **{**Steiner Neto**}**' or 'Steiner Neto, Pedro Jose' so as to treat 'Steiner Neto' as the surname and 'Pedro Jose' as the given name.
  4. Some names contain words, like 'Jr.' or 'Junior', at the end preceded by comma, e.g., 'John Morton, Jr.' or 'Osiris Detro, Junior'. Such names also may be produced as having multi-word surnames stated above, e.g., 'Osiris **{**Detro, Junior**}**' or 'Detro, Junior, Osiris' to represent 'Osiris Detro, Junior'.
  5. Some references may not contain name of any person, but the name of a company or an agency, e.g., 'John Wiley & Sons, Inc.' or 'Brahmaputra Pvt. Ltd.'. Such names may be produced by inserting them in curly braces, e.g., '**{**John Wiley ****& Sons, Inc.**}**' or '**{**Brahmaputra Pvt. Ltd.**}**'.
  6. In the case of a multi-author reference, the names of every two authors are to be separated by the word 'and' without enclosing it in curly braces in any case stated above, e.g., two authors as 'Dilip Datta and Pankaj

`Kumar Nath`', three authors as '`Dilip Datta and Pankaj Kumar Nath and Saptarshi Dutta`', and so on. Note that the names of two authors should not be separated by a comma (as generally done). In that case, instead of two authors, they will be treated as a single author with the first author as the surname and the second author as the given name, e.g., '`Dilip Datta, Pankaj Kumar Nath`' will be treated as a single author with '`Dilip Datta`' as the surname and '`Pankaj Kumar Nath`' as the given name.

7. If a long list is to be truncated with the name(s) of the first or few author(s), '`and others`', may be added after that(those) name(s), which will be converted to '`et al.`'.

- ▷ **booktitle**: Title of the book or proceedings, in which the referred article was published.
- ▷ **chapter**: Serial number of the referred unit, like chapter, section, or part of a book.
- ▷ **edition**: Edition of a book or a manual, e.g., '`Second`' or '`2nd`'.
- ▷ **editor**: Names of the editors, to be inserted in the same way as **author**.
- ▷ **howpublished**: Type of publication in the case of a manual or miscellaneous reference.
- ▷ **institution**: Name of the Institute, which published the referred report.
- ▷ **journal**: Name of the journal, in which the referred article was published.
- ▷ **month**: Month of publication, e.g., '`Jan.`' or '`January`'.
- ▷ **note**: A short note on the referred document, e.g., the abstract of an article.
- ▷ **number**: Serial number of the journal, magazine, proceedings, or technical report.
- ▷ **organization**: Name of the organization, which organized or sponsored the event.
- ▷ **pages** : Serial numbers of the referred pages, e.g., a single page as '`70`', a range of pages as '`24--35`', selective nonconsecutive pages as '`18,25,32`', or not very clear pages as '`57+`'.
- ▷ **publisher**: Name of the publisher.
- ▷ **school**: Name of the Institute, in which the referred thesis was submitted.
- ▷ **series**: Name of the series of a book or a proceedings (in addition to the **title** of the referred article), e.g., '`Studies in Computational Intelligence`' or '`Lecture Notes in Computer Science`'.
- ▷ **title**: Title of the reference. In **title**, all the intermediate alphabets in the title of a reference are printed in lowercase. In order to protect, intermediate uppercase alphabets of a title, or the entire title, may be put in a separate pair of curly braces, e.g., `title={A {P}ractical {G}uide to \LaTeX}`, `title={{A Practical Guide to \LaTeX}}`, `title="A {P}ractical {G}uide to \LaTeX"`, or `title="{A Practical Guide to \LaTeX}"`.
- ▷ **type**: Type of the reference, e.g., '`Ph.D thesis`', or '`Chapter`' or '`Section`' in the case of a book, and '`Research work`' in the case of a Technical report.
- ▷ **volume**: Volume number of the journal, proceedings, or multi-volume book.
- ▷ **year**: Year of publication, e.g., `2016`.

According to above, an illustrative bibliographic reference database file is shown in Table 15.2. Note that the first entry in the argument of a reference-type

Table 15.2 BIBTEX program compatible bibliographic reference database

```
% mybib2.bib
@article{Datta-Figueira-2013,
  author = {Dilip Datta and Jos\'e Rui Figueira},
  title = {{A real-integer-discrete-coded differential evolution}},
  journal = {Applied Soft Computing},
  volume = {13},
  number = {9},
  pages = {3884--3893},
  year = {2013}
}

@book{Deb-2001,
  author = {Kalyanmoy Deb},
  title = {{Multi-Objective Optimization using Evolutionary Algorithms}},
  publisher = {John Wiley \& Sons Ltd.},
  address = {Chichester, England},
  year = {2001}
}

@inproceedings{Burke-etal-1996,
  author = {Edmund Burke and Dave Elliman and Peter Ford Rupert Weare},
  title = {{Examination Timetabling in British Universities - A Survey}},
  booktitle = {Proceedings of Practice and Theory of Automated Timetabling},
  publisher = {Springer},
  series = {Lecture Notes in Computer Science (LNCS)},
  editor = {Edmund K. Burke and Peter Ross},
  year = {1996},
  volume = {1153},
  pages = {76--90}
}

@mastersthesis{Datta-1998,
  author = {Dilip Datta},
  title = {{Optimal Shape Design System for Plates under Dynamic Loads}},
  school = {Indian Institute of Technology, Delhi},
  month = {December},
  year = {1998},
  note = {Master thesis}
}

@techreport{Colorni-etal-1992,
  author = {Alberto Colorni and Marco Dorigo and Vittorio Maniezzo},
  title = {{A Genetic Algorithm to Solve the Timetable Problem}},
  number = {90-060 revised},
  institution = {Politecnico di Milano, Italy},
  year = {1992}
}
```

command is the mandatory citation key of a reference (e.g., `Datta-Figueira-2013` or `Deb-2001`), followed by the reference-specific mandatory and optional fields detailing the reference (like **author**, **title**, and **year**). In a reference-type command, two fields including the citation key are separated by a comma.

15.2 Standard Bibliographic Styles of LaTeX

The formatting of the list of bibliographic references in the BibTeX program is controlled by an associated bibliographic style. There exist a number of standard and alternative bibliographic styles of LaTeX, some of which are shown in Table 15.3.

Table 15.3 Some standard bibliographic styles of LaTeX

LaTeX style	Function
plain	References are listed in alphabetic order of the surnames (last or family names) of authors, and labeled by Arabic numerals in `[]`, e.g., the first two references of Table 15.2 will be produced as follows: [1] Dilip Datta and José Rui Figueira. A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893, 2013. [2] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001.
unsrt	Same with **plain**, except that the references are listed in order of their citations in the document.
alpha	A reference is labeled by an identifier generated from the surnames of the authors and the year of publication, and the references are listed in alphabetic order of their identifiers, e.g., the first two references of Table 15.2 will be produced as follows: [Deb01] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001. [DF13] Dilip Datta and José Rui Figueira. A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893, 2013.
abbrv	Same with **plain**, except that a reference is made compact by abbreviating the given (or first and middle) names of authors, e.g., the first two references of Table 15.2 will be produced as follows: [1] D. Datta and J. R. Figueira. A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893, 2013. [2] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001.
acm	Same with **plain**, but the surname of an author is printed first in small capital letters, followed by the abbreviated given name, e.g., the first two references of Table 15.2 will be produced as follows: [1] DATTA, D., AND FIGUEIRA, J. R. A real-integer-discrete-coded differential evolution. *Applied Soft Computing 13*, 9 (2013), 3884–3893. [2] DEB, K. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001.
apalike	The surname of an author is printed first, followed by the abbreviated given name, and a reference is labeled by an identifier generated from the surnames of authors and the year of publication, e.g., the first two references of Table 15.2 will be produced as follows: [Datta and Figueira, 2013] Datta, D. and Figueira, J. R. (2013). A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893. [Deb, 2001] Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England.

The style of bibliography is defined through the **\bibliographystyle{**`astyle`**}** command, where `astyle` is a bibliography style as given in Table 15.3. It is followed by the **\bibliography{**dbib**}** command, where dbib is the name of the bibliography database file without its .bib extension (multiple database files can also be loaded as **\bibliography{**dbib1,dbib2,...**}**). For example, following are the required commands to load the bibliography database file 'mybib2.bib' of Table 15.2 in **plain** style:

```
\bibliographystyle{plain}
\bibliography{mybib2}
```

Under all the bibliography styles given in Table 15.3, a reference is cited through the **\cite{**ckey**}** command, where ckey is the citation key of the reference. Multiple references can also be cited through a single **\cite{}** command separating two citation keys by a comma[2], e.g., **\cite{**Datta-1998, Even-etal-1976**}**. Citations are marked in the contents of a document by the identifiers of the cited references, e.g., [1], [2], ..., or [Deb01], [DF13], ..., or [Datta and Figueira, 2013], [Deb, 2001], ..., as shown in Table 15.3.

Only those references, cited in the document, are printed in the bibliographic reference list under the heading 'References'. If a reference is to be printed in the bibliography list without citing it in the document, the **\nocite{}** command may be used somewhere in the document, e.g., **\nocite{**Datta-1998**}** for printing the reference under the citation key 'Datta-1998' without citing it in the document. On the other hand, the **\nocite{*}** command may be used for listing all the references of a database file without citing even a single one.

15.3 Use of the natbib Package

It maybe noticed in §15.2 that each reference under the standard LATEX bibliographic styles is listed and cited with an identifier, like [1], [DF13], or [Deb, 2001]. In many applications, however, it may be preferred to cite a reference in author-year mode, like 'Datta and Figueira (2013)' or '(Deb, 2001)'. In such cases, the **natbib** package may be used, so as to list the references without any identifier and cite them in author-year mode. For this purpose, the **natbib** package provides its own bibliographic styles, such as **plainnat**, **unsrtnat** and **abbrvnat**, which are **natbib**-compatible versions of the standard LATEX styles **plain**, **unsrt**, and **abbrv**, respectively. The functions of the bibliographic styles, defined in the **natbib** package, are explained in Table 15.4 on the next page.

Note that the **abbrvnat** style lists an author as the abbreviated given name, followed by the surname. Some applications, however, may prefer to print the surname first, followed by the abbreviated given name, as done by the standard LATEX style **apalike**. For such output, the **apalike** style maybe used under the **natbib** package, in which case the references will be listed as shown in Table 15.3 but without any identifier.

[2]Multiple references can be cited through a single **\cite{}** separating two citation keys by a comma.

Table 15.4 Some bibliographic styles defined in the **natbib** package

natbib style	**Function**
plainnat	References are listed in alphabetic order of the surnames of authors, e.g., the first two references of Table 15.2 will be produced as follows: Dilip Datta and José Rui Figueira. A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893, 2013. Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001.
unsrtnat	Same with **plainnat**, except that the references are listed in order of their citations in the document.
abbrvnat	Same with **plainnat**, except that references are made compact by abbreviating the given names of authors, e.g., the first two references of Table 15.2 will be produced as follows: D. Datta and J. R. Figueira. A real-integer-discrete-coded differential evolution. *Applied Soft Computing*, 13(9):3884–3893, 2013. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons Ltd., Chichester, England, 2001.

Not only own bibliographic styles, the **natbib** package provides its own citation commands also for marking citations in different formats, such as **\citet{}**, **\citep{}**, **\citeauthor{}**, **\citeyear{}**, **\citeyearpar{}**, **\citealt{}**, and **\citealp{}** (the **\cite{}** command is still applicable, which would be equivalent to **\citet{}**). Upon loading the package with the **round** option as **\usepackage[round]{natbib}**, the behaviors of these citation commands are shown in Table 15.5 with self-explanatory applications. It is to be

Table 15.5 Citation commands provided under the **natbib** package

Command	**Function**	**Type of output**
\citet{ddf08}	Truncated author list (i.e., the first surname with et al. for more than two authors) with year in parentheses	Datta et al. (2008)
\citet*{ddf08}	Full author list with year in parentheses	Datta, Deb, and Fonseca (2008)
\citep{ddf08}	Truncated author list and year, both in parentheses	(Datta et al., 2008)
\citep*{ddf08}	Full author list and year, both in parentheses	(Datta, Deb, and Fonseca, 2008)
\citeauthor{ddf08}	Truncated author list without year	Datta et al.
\citeauthor*{ddf08}	Full author list without year	Datta, Deb, and Fonseca
\citeyear{ddf08}	Only year	2008
\citeyearpar{ddf08}	Only year (in parentheses)	(2008)
\citealt{ddf08}	Truncated author list and year	Datta et al. 2008
\citealp{ddf08}	Truncated author list and year, separated by a comma	Datta et al., 2008

mentioned that as in **\cite{}** stated in §15.2, multiple references may be cited through any of the commands of Table 15.4 also, e.g., **\citet{**ddf08,deb01**}** or **\citep{**ddf08,deb01**}**.

The patterns of citation markings under the **natbib** package can be altered by using different options to **\usepackage[]{}**. Some of such options are given in Table 15.6 on the next page. Provision is also there for obtaining combined effect using multiple options, separating two options by a comma, e.g., for obtaining compressed numbered citations in curly braces, the **natbib** package is to be loaded as **\usepackage[curly, numbers, sort&compress]{natbib}**. It is to be mentioned that, if the **numbers** option is used, the references will be listed by numbers.

Table 15.6 Citation patterns under the **natbib** package

	Option	Function (†Default)	Output of **\citet{}**, or **\citep{}**
Delimiter	**round**	Parentheses	Datta (2013), or (Deb, 2015)
	square	Square brackets†	Datta [2013], or [Deb, 2015]
	curly	Curly braces	Datta {2013}, or {Deb, 2015}
	angle	Angle brackets	Datta <2013>, or <Deb, 2015>
Separator of two citations	**colon**	Semi-colon†	Datta (2013); Deb (2015), or (Datta, 2013; Deb, 2001)
	comma	Comma	Datta (2013), Deb (2015), or (Datta, 2013, Deb, 2001)
Citation mode	**authoryear**	Author and year†	Datta (2013), or (Deb, 2015)
	numbers	Numbered	Datta [1], or [2,3,4,7]
	super	Superscribed	Datta^{1}, or ^{2,3,4,7}
Sorting	**sort**	Sorting as per reference list	Datta (2013), Deb (2015), or (Datta, 2013, Deb, 2001)
	sort&compress	Sorting as per reference list and compressing numeric citations	Datta [1], or [2–4,7]
First long	**longnamesfirst**	First citation in starred mode† and rest as instructed by users	Datta, Deb, Fonseca (2008), or (Datta et al., 2008)

15.4 Compiling BIBTEX based LATEX Input File

If the bibliography is generated through the BIBTEX program, the compilation of the LATEX file is to be changed from that addressed in §14.3 on page 139. In this case, it is to be compiled by using the following four lines of commands:

```
$ latex myarticle
$ bibtex myarticle
$ latex myarticle
$ latex myarticle
```

where '`myarticle`' is the name of the LATEX input file with '`tex`' extension. The 'bibtex' command compiles the bibliography file included in `myarticle.tex`. The last two 'latex' commands link the BIBTEX generated bibliographic references with LATEX. The above four lines of commands will produce five files, namely `myarticle.aux`, `myarticle.log`, `myarticle.dvi`, `myarticle.bbl`, and `myarticle.blg` (§20.4.1 on page 199 discusses in detail). As mentioned in §1.4 on page 4, the '`myarticle.dvi`' file can be viewed in a document viewer or can be used to produce a '`.ps`' or '`.pdf`' file.

15.5 Editing the .bbl File*

As stated at the beginning of this Hour that the BIBTEX program internally prepares the list of references in the **thebibliography** environment, it is stored in a `.bbl` file (refer §15.4). If the BIBTeX generated bibliographic reference list is not satisfactory, the required changes can be made in the `.bbl` file. For this, the following four steps are to be followed:

1. Save the `.bbl` file with another name, e.g., `edbbl.bib`. The change of name is necessary, otherwise the previous `.bbl` file (i.e., the edited `.bbl` file) will be overwritten by the new one if recompiled using the commands stated in §15.4.
2. Make the necessary changes in the `edbbl.bib` file according to the instructions addressed in §14.1 on page 137 for preparing the database under the **thebibliography** environment.
3. In the LaTeX input file, replace the **\bibliographystyle{}** and **\bibliography{}** commands by the **\input{**`edbbl.bib`**}** command (note that, as stated in §15.2, the original bibliographic database file is linked in the LaTeX input file through **\bibliographystyle{}** and **\bibliography{}**). Some publishers may ask to put the bibliographic references in the LaTeX input file itself, instead of putting in a separate file. In that case, instead of the **\input{**`edbbl.bib`**}** command, the contents of the `edbbl.bib` file may simply be copied and pasted in that location.
4. Recompile the LaTeX input file as stated in §14.3 on page 140 or §15.4 on the previous page.

15.6 Multiple Bibliographies*

In a document, in which different units (like chapters of a book, or sections of an article) are prepared in separate `.tex` files and then included in a root file using the **\include{}** command (§20.2 on page 192 discusses in detail), unit-wise separate bibliographic reference lists can be produced with the support of the **chapterbib** package. This provision may be required specially in edited books or conference proceedings, which contain units written by different authors. All the bibliographic styles and citation commands discussed above, as well as in Hour 14, are still applicable. Additionally, the following jobs will be required:

1. Load the **chapterbib** package in the preamble.
2. At the end of each unit file, insert the required bibliographic reference database, as follows, either in the **thebibliography** environment as stated in Hour 14 or through the **\bibliographystyle{}** and **\bibliography{}** commands as stated in §15.2.

    1. **\begingroup**
    2. **\let\clearpage\relax**
    3. **\renewcommand{\bibname}{**`References`**}**
    4. **\vskip** `5mm`
    5. Either **\begin{thebibliography}{}** … **\end{thebibliography}**
       or **\bibliographystyle{} \bibliography{}**, as applicable
    6. **\endgroup**

In order to implement the optional local effects of lines 2–4 (particularly under some document-classes, like **book** or **report**), the commands are grouped by the **\begingroup** and **\endgroup** commands in lines 1 and 6, respectively. The set of commands in line 2 removes intermediate blank pages, if any, and starts the bibliographic reference list on the same page with the contents of the unit. The command in line 3 is to produce the bibliographic list under the heading '`References`', while the command in line 4 is to maintain some vertical blank space (manually set) on the top of the heading '`References`'. Finally, the mandatory **thebibliography** environment or the set of **\bibliographystyle{}** and **\bibliography{}** commands is to insert the required bibliographic reference database.

3. Compile the files according to §14.3 on page 140 if the reference database is inserted in the **thebibliography** environment, while according to §15.4 if it is produced through BIBTEX. In the case of BIBTEX, each unit file is to be compiled with a separate bibtex command. As an example, consider that two unit files, `myunit1.tex` and `myunit2.tex`, are included in the root file `mydoc.tex`. Then these are to be compiled through the following five lines of commands:

```
$ latex mydoc
$ bibtex myunit1
$ bibtex myunit2
$ latex mydoc
$ latex mydoc
```

Hour 16

Lists of Contents and Index

Lists of contents and index are generally prepared in a big document, such as a book or a report. The lists of contents are prepared at the beginning of a book showing page-wise headings of various topics and captions of tables and figures, while the index prepared at the end shows the page numbers of topic-related various terms covered in the book.

16.1 Lists of Contents

LaTeX provides the **\tableofcontents**, **\listoftables** and **\listoffigures** commands for automatic generation of three lists in a document, which are contents of sectional units, contents of tables, and contents of figures, respectively. These three lists are produced under the headings of '`Contents`', '`List of Tables`', and '`List of Figures`', respectively. The **\tableofcontents**, **\listoftables**, and **\listoffigures** commands are to be placed in their proper locations in the **document** environment, generally in between the front matter and main matter of a document (detail is in §20.1 on page 191).

16.1.1 Information to the Lists of Contents

Contents of the list of '`Contents`' are the arguments of various sectional commands (like **\chapter{}**, **\section{}**, etc.), while those of '`List of Tables`' and '`List of Figures`' are the arguments of the **\caption{}** command used in different tables and figures, respectively.

Numbered sectional and captioned items, like **\section{}** or **\caption{}**, are automatically included in the respective lists of contents using their arguments. If it happens that the mandatory argument of a command is too long to include in its list of contents, provision is there for printing a shorter information in the list. This is done through the optional argument of a sectional or caption command,

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_16

like \chapter[]{}, \section[]{} or \caption[]{}[1]. For example, if the caption of a table is written as \caption[caplot]{captab}, then the table in its position will be captioned with the mandatory captab, while it will be included in the 'List of Tables' with the optional caplot (in the absence of the optional caplot, however, it will be listed with the mandatory captab). Such examples can be found in many sectional units, tables, and figures of this book.

On the other hand, unnumbered sectional and captioned items, i.e., an item with the starred form of a command, like \section*{} or \caption*{}, are excluded from a list of contents. Such an item, however, can be included in a list of contents through the \addcontentsline{}{}{} command as \addcontentsline{alist}{atype}{info}, where alist is the list of contents in which the item is to be included (toc for 'Table of Contents', lot for 'List of Tables' and lof for 'List of figures'), atype is the type of the item (e.g., chapter, table, or figure), and info is the information to be entered in the list of contents. For example, the *Preface* of a book is usually written through the \chapter*{} command as a chapter without any serial number, which can be included in the list of 'Contents' by inserting the \addcontentsline{toc}{chapter}{Preface} command just before the inclusion of the input file containing the *Preface* of the book (refer §20.2 on page 192 for detail). Similarly, as done in this book, the unnumbered lists generated by the \tableofcontents, \listoftables, and \listoffigures commands can be included in the list of 'Contents' through the \addcontentsline{toc}{chapter}{} command with the last argument as the heading to be entered in the list of 'Contents' (detail is in §20.2). Any fragile command in the last argument of \addcontentsline{}{}{} should be protected through the \protect command (§18.6 on page 176 discusses fragile commands).

If an item, included in a list of contents through \addcontentsline{}{}{} as above, is shown on a wrong page in the list of contents (usually the previous page of its actual occurrence), \phantomsection command defined in the hyperref package may be added before \addcontentsline{}{}{}, e.g., \phantomsection\addcontentsline{toc}{chapter}{Preface}. In the case of two such consecutive items, the former item may need to be ended additionally by \clearpage.

*16.1.2 Formatting Lists of Contents**

Some provisions for altering the default formatting of the lists of contents:

1. The headings produced by \tableofcontents, \listoftables, and \listoffigures can be changed by redefining, respectively, the \contentsname, \listtablename, and \listfigurename commands through the \renewcommand{}{} command as discussed in §13.2 on page 128, e.g., '\renewcommand{\contentsname}{List of Contents}' will replace the default heading 'Contents' of \tableofcontents with 'List of Contents'.

---

[1]If the heading/title of a numbered unit (like \chapter{}, \section{}, or \caption{}) is too long to include in lists of contents, a shorter heading/title can be generated through an optional argument of the command.

2. By default all the numbered sectional units are listed in the `Contents` of a document. This may increase the size of the `Contents`, particularly when the depth of sectional units is increased as mentioned in §4.1 on page 27. In that case also, however, the size of the `Contents` can be controlled by suppressing some lower level of sectional units. This suppression is accomplished through the **tocdepth** counter. For example, **\setcounter{tocdepth}{**`2`**}** (in the preamble) will restrict the list of `Contents` of a document only up to the third level of sectional units, e.g., **\part{}**, **\chapter{}**, and **\section{}** in the case of a book divided into parts, or **\chapter{}**, **\section{}**, and **\subsection{}** in a book without any part. Similar to **tocdepth**, the levels of depth to be incorporated in the `List of Tables` and `List of Figures` can be increased or decreased by changing the values of **lotdepth** and **lofdepth**, respectively.
3. Notice in the lists of contents on pages (ix)–(xxvii) of this book that the gap between an entry and its page number is filled by dots. The gap between two dots can be adjusted by redefining the value of the **\@dotsep** command, e.g., **\renewcommand{\@dotsep}{**`3.5`**}**. A very large value may be assigned to **\@dotsep** if dots are not required at all, e.g., **\renewcommand{\@dotsep}{**`500`**}** will eliminate the dots in the lists of contents of this book.
4. Notice on pages (ix)–(xxvii) of this book that the page numbering column in each of the `Contents`, `List of Tables`, and `List of Figures` has the heading of 'Page'. It is not default, but produced using the **\addtocontents{**`alist`**}{~\hfill** `Page`**\par}** command three times with values of `alist` as **toc**, **lot**, and **lof**, respectively. The commands are inserted at the start of the **document** environment, before inserting any unit that will be included in the lists of contents.
5. Further notice in the `List of Tables` and `List of Figures` on pages (xvii)–(xxvii) of this book that the entries (i.e., captions) of each Hour is preceded by the corresponding Hour heading. It is also not default, but produced by inserting **\addcontentsline{lot}{chapter}{\thechapter** `atitle`**}** and **\addcontentsline{lof}{chapter}{\thechapter** `atitle`**}** just after the **\chapter{}** command of each applicable Hour, where `atitle` is the heading of the Hour.
6. By default the appendices in the list of `Contents` are marked by uppercase alphabets, like `A`, `B`, etc. However, the only appendix of this book is marked as '`Appendix A`', i.e., '`A`' is preceded by the word '`Appendix`'. This is obtained by including the appendix file in the root file (Table 20.5 on page 196 discusses in detail) through the **appendices** environment as follows:

```
\begin{appendices}
\renewcommand{\chaptername}{Appendix}
\include{appsymb}
\end{appendices}
```

   Since the appendix file (named as `appsymb.tex`) is prepared under the **\chapter{}** command, the **\chaptername** command is renamed as '`Appendix`' before including the appendix file. The **appendices** environment is defined in the **appendix** package, which is loaded in the preamble as **\usepackage[titletoc]{appendix}** instructing through the **titletoc** option to put the title of the appendix in the list of `Contents`.

16.1.3 Multiple Lists of Contents*

Sometime the list of `Contents` in a book may need to be generated chapter-wise, instead of a single global one or in addition to that. This may be required particularly in an edited book or a conference proceedings, where its chapters are written by different authors. Such chapter-wise lists of `Contents` are generated through the **minitoc** package.

A sample LaTeX input file for an edited book with chapter-wise lists of `Contents` is shown in Table 16.1 along with its output in Table 16.2 on the next page.

Table 16.1 Book with chapter-wise lists of `Contents`

```
\documentclass[a4paper,openany]{book}
\usepackage{minitoc}
\setcounter{tocdepth}{0}
\setcounter{minitocdepth}{2}
%
\begin{document}
\dominitoc
\tableofcontents
% \faketableofcontents
%
\chapter[Introduction to \LaTeX\\{\it Dr.\␣D.\␣Datta}]{Introduction to \LaTeX}
\begin{flushright}  Dr.\␣D.\␣Datta  \end{flushright}
\minitoc
\section{What is \LaTeX?}
\LaTeX\ is a macro package ...
\subsection{Commands and environments}
A document is prepared by interspersing ...
\subsection{How to write a document?}
The simplest \LaTeX\ document is started ...
%
\chapter[Basic Formatting Tools\\{\it Dr.\␣P.\␣Das}]{Basic Formatting Tools}
\begin{flushright}  Dr.\␣P.\␣Das  \end{flushright}
\minitoc
\section{Formatting page size}
The $\backslash$documentclass[]\{\} command allows ...
\section{Formatting page numbering}
As seen in the figure ...
\section{Sectional units}
Various sectional units ...
%
\chapter[Mathematical Expressions\\{\it Dr.\␣R.\␣Rana}]{Mathematical Expressions}
\begin{flushright}  Dr.\␣R.\␣Rana  \end{flushright}
\minitoc
\section{Mathematical notations}
Different mathematical notations are the basic tools for writing mathematical expressions ...
\section{Mathematical operators}
In order to form a mathematical expression, various terms are connected by some operators ...
\end{document}
```

Table 16.2 Book with chapter-wise lists of `Contents` (output of the input file of Table 16.1)

Contents

1

Chapter 2

Basic Formatting Tools

Dr. P. Das

Contents

2.1 Formatting page size

The \documentclass[]{ } command allows ...

2.2 Formatting page numbering

As seen in the figure ...

2.3 Sectional units

Various sectional units ...

3

Chapter 1

Introduction to LaTeX

Dr. D. Datta

Contents

1.1 What is LaTeX?

LaTeX is a macro package ...

1.1.1 Commands and environments

A document is prepared by interspersing ...

1.1.2 How to write a document?

The simplest LaTeX document is started ...

2

Chapter 3

Mathematical Expressions

Dr. R. Rana

Contents

3.1 Mathematical notations

Different mathematical notations are the basic tools for writing mathematical expressions ...

3.2 Mathematical operators

In order to form a mathematical expression, various terms are connected by some operators ...

4

The **\setcounter{tocdepth}**{0} command is used in the LaTeX input file of Table 16.1 for producing a global list of `Contents` only with the chapter headings of the book. On the other hand, the **\setcounter{minitocdepth}**{2} command will generate a list of Contents for a chapter covering up to the second level of sectional units, i.e., up to the \section{} and \subsection{} commands. The \dominitoc command, used immediately after starting the document environment, is a mandatory command for initializing the minitoc system. The next command is \tableofcontents for producing the global list of Contents. If the global list of Contents is not to be produced, the \tableofcontents command should be replaced by the \faketableofcontents command, which is shown commented in Table 16.1. Then the actual contents of the book is inserted as usual, with the only difference that each \chapter{} command is followed by the \minitoc command for producing a list of Contents for that chapter. Since the example is for an edited book, whose chapters are usually written by different authors, the author of each chapter is inserted through the flushright environment between its \chapter{} and \minitoc commands. Moreover, the name of the author is also included in the optional argument of the \chapter[]{} command for printing it in the global list of Contents, if it is opted.

16.1.4 Compiling LaTeX Input File Having Lists of Contents

A LaTeX input file, which is to produce lists of contents also, is to be compiled twice through the latex command as stated in §14.3 on page 140. The first latex command compiles the LaTeX input file and generates some additional files related to the lists of contents, while the second latex command links the generated lists of contents with LaTeX. If only the global lists of contents are to be produced, the compilation generates four files with dvi, aux, log, and toc extensions. The compilation of the minitoc system generates some more files, one with bmt extension, one with mtc extension, and one with mtc<N> extension against each chapter, where <N> is the serial number of a chapter.

16.2 Making Index

The LaTeX command for indexing a term is \index{}, which is defined in the makeidx package. The index of a document is always produced on a new page under the heading 'Index'. In order to produce the index in the output, besides loading the makeidx package, the \makeindex and \printindex commands are also required to be inserted in the LaTeX input file. The \makeindex command is inserted in the preamble (after all the \usepackage{} commands) instructing to prepare index, while \printindex is inserted in the body of the document (usually just before the \end{document} command) for producing the index in the output (refer Table 16.3 on the next page). A term in the index list in the output is followed by the page number of the document on which the term appears.

Table 16.3 Document having index

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td rowspan="2">\documentclass[a4paper,12pt]{article}
\usepackage{makeidx}
\makeindex
%
\begin{document}
Dynamics\index{Dynamics} is a part of
mechanics\index{Mechanics}. It is divided into
kinematics and kinetics. The
kinematics\index{Dynamics!Kinematics} deals with
motions\index{Dynamics!Kinematics!Motion} only, while
the kinetics\index{Kinetics}
\index{Kinetics!|seealso{Kinematics}} deals with both
motions and forces\index{Force|see{Kinetics}}.
%
\printindex
\end{document}</td><td>Dynamics is a part of mechanics. It is divided into kinematics and kinetics. The kinematics deals with motions only, while the kinetics deals with both motions and forces.

1</td></tr>
<tr><td>Index

Dynamics, 1
  Kinematics, 1
    Motion, 1
Force, see Kinetics
Kinetics, 1, see also Kinematics
Mechanics, 1

2</td></tr>
</table>

16.2.1 Indexing Terms

The term to be indexed is immediately followed by the **\index{}** command with the term or its required form as the argument of the command. In order to avoid access blank space in the output, there should not be any gap between the term to be indexed and the **\index{}** command, e.g., ‘`composite\index{Composite}`’ for indexing the word ‘`composite`’ by its capitalized form ‘`Composite`’.

A maximum of three-tier index can be prepared using a **!** sign before each sub-index. These are shown in Table 16.3 through `\index{Dynamics}`, `\index{Dynamics!Kinematics}` and `\index{Dynamics!Kinematics!Motion}` for indexing ‘`Dynamics`’ alone (first tier), ‘`Kinematics`’ under ‘`Dynamics`’ (second tier), and ‘`Motion`’ under ‘`Kinematics`’ which is again under ‘`Dynamics`’ (third tier), respectively (see their printing formats in Table 16.3).

An index can also be referred to another index using the **see{}** or **seealso{}** command, whose argument is the referred index. The **see{}** command (used for `Kinetics` in Table 16.3) is preceded by a **|**, while the **seealso{}** command (used for `Kinematics` in Table 16.3) is preceded by **!|**. Note that the word ‘`Kinetics`’ is indexed twice, the first one is for putting the page number, and the second one is for referring it to the word ‘`Kinematics`’ through the **seealso{}** command.

16.2.2 Some Guidelines on Indexing

The following are some guidelines to be obeyed while indexing a term:

1. Special LaTeX characters, like `#`, `$`, `%`, or `&`, are to be indexed in the same way as they are produced in the body of a document, e.g., **\index{\$}** for indexing `$`.
2. To index a character having a special meaning to **\makeindex**, such as `!`, `"`, `@`, or `|`, the character is to be preceded by **"** in the argument of **\index{}**[2]. For example, **\index{**"`@article`**\{\}}** for indexing `@article{}`, **\index{"!$"|$}** for indexing `!|`, or **\index{**\verb"" ""**}** for indexing \verb" ".
3. If the term to be produced in the index list is not exactly the same with the argument of **\index{}** (i.e., if the indexed term is to be produced through another LaTeX command), the other command may be preceded by **@**, e.g., **\index{**`Boldface`**@\textbf{**`Boldface`**}}** to index `Boldface` as **Boldface**, or **\index{**`sigma`**@σ}** for indexing `sigma` as σ. Without **@**, the entry in the index list will be alphabetized wrongly, e.g., σ as **σ** in the previous example.
4. To include a range of pages for an indexed term, the **\index{**`aterm`**|(}** command is to be used at the beginning and **\index{**`aterm`**|)}** at the end of the range, where `aterm` is the term to be indexed.

16.2.3 Compiling a LaTeX Input File Having Index

Like the bibtex command used for compiling a document with a BibTeX based bibliographic reference list (refer §15.4 on page 149), the makeindex command is required for compiling a document with an index list. In this case, a LaTeX file is to be compiled using the following four lines of commands:

```
$ latex myarticle
$ makeindex myarticle
$ latex myarticle
$ latex myarticle
```

where '`myarticle`' is the LaTeX input file with '`tex`' extension. The 'makeindex' command compiles the **\index{}** commands included in `myarticle.tex`. The last two 'latex' commands link the generated list of index with LaTeX. The above four lines of commands will produce six files, namely `myarticle.aux`, `myarticle.log`, `myarticle.dvi`, `myarticle.idx`, `myarticle.ilg`, and `myarticle.ind` (§20.4.1 on page 199 discusses in detail). Out of these six files, `myarticle.dvi` can be used for producing a '`.ps`' or a '`.pdf`' file as mentioned in §1.4 on page 4.

---

[2]To index a character having a special meaning to **\makeindex**, such as `!`, `"`, `@`, or `|`, the character is to be preceded by **"** in the argument of **\index{}**.

Hour 17

Miscellaneous I

Previous Hours were devoted on various issues related to the preparation of a general document. This Hour discusses about some special effects that can be produced in a document, such as important notes and equations in boxes, geometric transformation, etc.

17.1 Boxed Items

Important pieces of information can be produced in different types of boxes in order to make them prominent, which are discussed in this section.

17.1.1 Texts in Plain Boxes

Various commands for printing a single-line texts in a box include **\frame{}**, **\framebox{}**, **\fbox{}**, **\doublebox{}**, **\ovalbox{}**, **\Ovalbox{}**, **\shadowbox{}**, **\shabox{}**, etc. Details of these commands are shown in Table 17.1 on the next page, in which the types of boxes produced under different commands may be noticed. The commands of Table 17.1 can be used in running texts, e.g., **\fbox{**`boxed note`**}** prints boxed note in this line.

The width of lines and space for starting contents in the boxes produced by the commands of Table 17.1 (excluding **\frame{}** and **\shabox{}**) can be controlled by the **\fboxrule{}** and **\fboxsep{}** commands, e.g., **\setlength{\fboxrule}{**2**pt}** for producing a box of line width of 2 pt (default is 0.4 pt) and **\setlength{\fboxsep}{**5**pt}** for printing the contents at a distance of 5 pt (default is 3 pt) from all the lines of a box. Further, the commands of Table 17.1 can be nested for a combined effect, e.g., **\fbox{\fbox{**`double boxes`**}}** for producing double boxes.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_17

Table 17.1 Single-line texts in boxes

Command	Package	LaTeX input	Output
\frame{}	–	\frame{A single-line box}	A single-line box
\framebox{}	–	\framebox{A single-line box}	A single-line box
\fbox{}	–	\fbox{A single-line box}	A single-line box
\doublebox{}	fancybox	\doublebox{A double-line box}	A double-line box
\ovalbox{}	fancybox	\ovalbox{An oval box}	An oval box
\Ovalbox{}	fancybox	\Ovalbox{A thick oval box}	A thick oval box
\shadowbox{}	fancybox	\shadowbox{A shaded box}	A shaded box
\shabox{}	shadow	\shabox{A big shaded box}	A big shaded box

Note that the arguments of the commands of Table 17.1 are printed in LR-mode (from left to right) in a single line, and the length of a box is calculated automatically according to the size of the argument of a command. If a box of a user-specified length or different alignments of the argument are required, the \framebox{} command may be used with two optional arguments, i.e., as \framebox[alen][algn]{acont}, where acont is the contents to be printed with algn alignment in the box of length alen. The available alignment options are l, c, r, and s, applied respectively for left-aligned, centered (default), right-aligned, and stretching full length of the box. For example, \framebox[8cm][r]{Box of user-defined length and alignment} will print Box of user-defined length and alignment. Similar to \framebox[][]{}, the \makebox[alen][algn]{acont} command may also be used, in which however the box remains invisible. In a special application, particularly in the picture environment (refer §10.5 on page 97 for detail), \makebox[][]{} may be applied with zero length for printing texts in a particular position. The command may also be used for printing overlapping texts, e.g., '\makebox[0mm][l]{-----}CUT' will print '~~CUT~~', or \makebox[0mm][l]{/}L will produce Ł.

17.1.2 Texts in Color Boxes

The \colorbox{bcol}{atext} and \fcolorbox{brcol}{bcol}{atext} commands are defined in the color package for printing texts in colored boxes, where atext is the texts to be produced in the box, bcol is the background color of the box and brcol is the border color of the box. For example, \colorbox{black}{\textcolor{white}{White texts in black box}} and

`\fcolorbox{red}{black}{\textcolor{white}{White texts in black box with red border}}` will produce, respectively, White texts in black box and White texts in black box with red border.

The **color** package provides the `\pagecolor{}` command also (e.g., `\pagecolor{green}`), which changes the background color of all the remaining pages of a document starting from the current page.

17.1.3 Mathematical Expressions in Boxes

The direct command for producing mathematical expressions in boxes is `\boxed{}` defined in the **amsmath** package, whose argument is processed in math-mode as shown in the first example in Table 17.2. The commands of Table 17.1, as well as

Table 17.2 Equations in boxes

LaTeX input	Output
`\boxed{x^2 + y^2 = r^2}`	$\boxed{x^2+y^2=r^2}$
`\shabox{$x^2 + y^2 = r^2$}`	$\boxed{x^2+y^2=r^2}$
`\begin{equation}` `\boxed{x^2 + y^2 = r^2}` `\end{equation}`	$\boxed{x^2+y^2=r^2}$ (17.1)
`\begin{equation}` `\shabox{$x^2 + y^2 = r^2$}` `\label{eq_in_box}` `\end{equation}`	$\boxed{x^2+y^2=r^2}$ (17.2)

`\colorbox{}{}` and `\fcolorbox{}{}{}`, can also be used for producing mathematical expressions in boxes. However, since the arguments of these commands are processed in text-mode, a mathematical expression is to be inserted in an inline math-mode (refer §11.3 on page 104 for detail), say in a pair of `$` as shown in the second example in Table 17.2 for producing an equation through `\shabox{}`. If the equation in a box is to be numbered (which can be referred also through a reference key), the `\boxed{}` or `\shabox{}` command may be put in the **equation** environment as shown in the third and fourth examples in Table 17.2.

*17.1.4 Paragraphs in Boxes**

One drawback with the commands of Table 17.1 is that the entire argument of a command is printed in a single line without any line break, even continuing beyond the width of a page. A new line or a line break command (\newline or \\) is also not accepted by these commands. Hence, a long piece of texts may be produced through the \parbox[valgn]{ahorz}{atext} command, which prints atext in an invisible box of ahorz length with optional valgn for vertical alignment, with proper line breaking if required. The permitted vertical alignments of the box of \parbox[]{}{} include t for top alignment, c (default) for centered, and b for bottom alignment. In order to produce a visible box, \parbox[]{}{} may be put in a command given in Table 17.1.

Some applications of \parbox[]{}{} are shown in Table 17.3. Note that, if ahorz is not sufficient to hold a word, it may go even beyond the box produced by \parbox[]{}{}, which is demonstrated in the first example in Table 17.3. Hence, as shown in the second example, \hspace{0pt} is used before atext for automatic hyphenation of a long word to accommodate it within the box. The third example in Table 17.3 shows how a long atext is produced in a full-justified paragraph with automatic line break, while the fourth example shows that the manually set line break command '\\' is also accepted by \parbox[]{}{}. Further, the optional vertical alignment of a box produced by \parbox[]{}{} may also be noticed in the third and fourth examples in Table 17.3 (top aligned in the third example and centered in the fourth example).

Table 17.3 Paragraphs in boxes through the \parbox[]{}{} command

LaTeX input	Output
`\fbox{\parbox{17mm}{Characteristics}}`	Characteristics
`\fbox{\parbox{17mm}{\hspace{0pt}Characteristics}}`	Character- istics

`See it \fbox{\parbox[t]{2.8cm}{\hspace{0pt}`
`Characteristics may be studied in a systematic way.}}`
`carefully.`
	See it Characteristics may be studied in a sys- tematic way. carefully.

`See it \fbox{\parbox[c]{2.8cm}{\hspace{0pt}`
`Characteristics may be studied in a\\systematic`
`way.}} carefully.`
	See it Characteristics may be studied in a systematic way. carefully.

17.1.5 Set of Items in a Box

It is discussed in §17.1.4 how a long item in a box can be printed in multiple lines in the form of a paragraph with automatic line breaking. However, that process cannot be applied conveniently for printing a set of items in a single box, like an array of equations or some pointed items. The simplest process for such a requirement is to use the boxedminipage environment defined in the boxedminipage package. It is similar with the minipage environment (refer §4.4 on page 31 for detail), with the only difference that the mini page under the boxedminipage environment is enclosed in a box.

Moreover, the commands of Table 17.1 accept some environments in their arguments, which also facilitate to print different items in individual lines in a single box.

Table 17.4 Array of equations in a box through the **tabular** environment in **\shabox{}**

LaTeX input	Output

`\begin{equation}`
`\shabox{`
`\begin{tabular}{l}`
`$s = vt$\\`
`$v = v_0 + at$\\`
`$v^2 = v_0^2+ 2as$`
`\end{tabular}`
`}`
`\end{equation}`
	$\boxed{\begin{array}{l} s = vt \\ v = v_0 + at \\ v^2 = v_0^2 + 2as \end{array}}$ (17.3)

Table 17.4 shows an array of equations produced through a single-column **tabular** environment as the argument of **\shabox{}**, which is put in the **equation** environment for assigning a serial number to the equations.

If the array of equations of Table 17.4 are to be aligned or numbered individually, the **Beqnarray** environment, defined in the **fancybox** package, may be used instead of the **tabular** environment. The **Beqnarray** is a math-mode environment and it is similar with the **eqnarray** environment, the only difference is that the former can be used as the argument of a command of Table 17.1 for producing an array of equations in a box. An application of the **Beqnarray** environment is shown in Table 17.5, where it is

Table 17.5 Array of equations in a box through the **Beqnarray** environment in **\shabox{}**

LaTeX input	Output

`\shabox{`
`\begin{Beqnarray}`
`s &=& vt \\`
`v &=& v_0 + at \\`
`v^2 &=& v_0^2+ 2as`
`\end{Beqnarray}`
`}`
	$s = vt$ (17.4)
	$v = v_0 + at$ (17.5)
	$v^2 = v_0^2 + 2as$ (17.6)

applied as the argument of **\shabox{}**. On the other hand, if none of the equations is to be numbered, the **Beqnarray** environment may be replaced by the **Beqnarray*** environment.

The **fancybox** package defines some more environments, such as **Bcenter**, **Bflushleft**, **Bflushright**, **Benumerate**, **Bitemize**, and **Bdescription**. The **Bcenter**, **Bflushleft**, and **Bflushright** environments are similar with the **center**, **flushleft**, and **flushright** environments (refer §3.3 on page 18 for detail), which are used for making a paragraph center-aligned, left-aligned, and right-aligned, respectively. On the other hand, the **Benumerate**, **Bitemize**, and **Bdescription** are similar, respectively, with the **enumerate**, **itemize**, and **description** environments (refer §6.1 on page 49 for detail), which are used for producing different listed items. The main advantage of these environments of the **fancybox** package is that they can be used in a box producing command for printing their contents in a box. However, they suffer from a drawback also, that they do not have any automatic line braking facility, for which a line may continue even beyond the margin of a page. To be within the page margin, a line break is to be provided manually. As an example, Table 17.6 shows the use of the **Bitemize** environment in **\shabox{}**.

Table 17.6 Unnumbered list in a box through the **Bitemize** environment in **\shabox{}**

LaTeX input	**Output**

`\shabox{`
`\begin{Bitemize}`
`\item Beqnarray produces an array of\\`
`    equations, similar to the\\`
`    eqnarray environment.`
`\item Benumerate, Bitemize and\\`
`    Bdescription produce different\\`
`    types of listed items, similar to\\`
`    the enumerate, itemize and\\`
`    description environments respectively.`
`\item Bcenter, Bflushleft and Bflushright\\`
`    make a paragraph center-aligned,\\`
`    left-aligned and right-aligned\\`
`    respectively, similar to the center,\\`
`    flushleft and flushright environments.`
`\end{Bitemize}`
`}`
	• Beqnarray produces an array of
	equations, similar to the
	eqnarray environment.
	• Benumerate, Bitemize and
	Bdescription produce different
	types of listed items, similar to
	the enumerate, itemize and
	description environments respectively.
	• Bcenter, Bflushleft and Bflushright
	make a paragraph center-aligned,
	left-aligned and right-aligned
	respectively, similar to the center,
	flushleft and flushright environments.

17.2 Rotated Items*

It is discussed in §7.4 on page 62 that a piece of texts or a table can be rotated by 90° in the counter-clockwise direction through the **sideways** environment. In a general case, the **rotate** environment, defined in the **rotating** package, can be used for rotating the contents of the environment by any amount specified as its mandatory argument in the form **\begin{rotate}{**`adeg`**}**, where `adeg` is the angle (in degree) by which the content is to be rotated (the **sideways** environment is a special case of the **rotate** environment for rotating by 90°). A positive value of `adeg` rotates the contents in the counter-clockwise direction, while a negative value rotates in the clockwise direction. Some applications of the **rotate** environment are given in Table 17.7.

Table 17.7 Rotated items through the **rotate** environment

LaTeX input	**Output**

`\begin{rotate}{30}`
`  Rotated by 30 degree.`
`\end{rotate}`
	Rotated by 30 degree.

`\begin{rotate}{-20}`
`  \fbox{Rotated by -20 degree.}`
`\end{rotate}`
	$\boxed{\text{Rotated by -20 degree.}}$

`The item rotated through the rotate environment …`
`Such an example is shown here,`
`\begin{rotate}{30}`
`  \fbox{\bf Rotated item.}`
`\end{rotate},`
`which got overlapped with some lines.`
	The item rotated through the rotate environment has the drawback that it may overlap with the contents of a document. Such an example is shown here, **Rotated item.**, which got overlapped with some lines.

An item rotated by the **rotate** environment can easily be put in a box using any command of Table 17.1, which is demonstrated in the second example in Table 17.7. As shown in the third example in Table 17.7, however, the **rotate** environment suffers from the drawback that a rotated item may get overlapped with other contents of a document. To avoid this drawback, an item may be rotated though the **turn** environment, instead of the **rotate** environment.

The **turn** environment, which is also defined in the **rotating** package, works exactly in the same way with that of the **rotate** environment, but prints its contents in a separate space without any overlapping. Some applications of the **turn** environment are given in Table 17.8. As in the case of the **rotate** environment, the contents of the **turn** environment also can be printed in a box as shown in the second and third examples in Table 17.8. Further, notice in these two examples that a rotated item can also be put in **\parbox[ ]{ }{ }** for printing it in the form of a paragraph with automatic line breaking. The effects of the optional vertical alignment of **\parbox[ ]{ }{ }** may also be noticed in these two examples (refer §17.1.4 for detail).

Table 17.8 Rotated items through the **turn** environment

LaTeX input	Output

`As shown with the help of an example here,`
`\begin{turn}{30}`
`  Rotated item.`
`\end{turn},`
`the turn environment prints its contents`
`without any overlapping.`
	As shown with the help of an example here, Rotated item. , the turn environment prints its contents without any overlapping.

`A long rotated item is printed in this example`
`\begin{turn}{30}`
`\fbox{\parbox[t]{2cm}{It is rotated by 30 degree.}}`
`\end{turn}`
`in multiple lines with automatic line breaking.`
	A long rotated item is printed in this example It is rotated by 30 degree. in multiple lines with automatic line breaking.

`A long rotated item is printed here`
`\begin{turn}{30}`
`\fbox{\parbox[b]{2cm}{It is rotated by 30 degree.}}`
`\end{turn}`
`in multiple lines with automatic line breaking.`
	A long rotated item is printed in this example It is rotated by 30 degree. in multiple lines with automatic line breaking.

17.3 Items at Different Levels and Forms*

Observe the word 'LaTeX', produced by the **\LaTeX** command, whose letters are printed in different heights, sizes, and spacing. Such patterns can be obtained through the **\raisebox{ }[ ][ ]{ }** command, in the form of

`\raisebox{ahgt}[atop][abot]{atext}` for raising (printing) `atext` at a height of `ahgt` from the current line (a negative value of `ahgt` will lower it) with optional `atop` and `abot` to create space, respectively, above and below the line for printing `atext` (`atop` is required while raising `atext`, and `abot` is required while lowering it).

Table 17.9 shows some applications of `\raisebox{}[][]{}`. In the first example, two words are raised at a height of 1 ex, which is accommodated with an optional vertical space of 3.5 ex above the line. Similarly, two words in the second example are lowered by 1 ex (i.e., raised at a height of -1 ex), which is accommodated with an optional vertical space of 2.5 ex below the line. Note that the first optional argument to `\raisebox{}[][]{}` (used to create a vertical space above the line) is also required in the second example, without which the second optional argument (used to create a vertical space below the line) will be treated as the first optional argument. However, since no extra vertical space above the line is required in this example, the first optional argument to `\raisebox{}[][]{}` is assigned the value of 0 ex. The third example in Table 17.9 is slightly different. No optional argument is used in `\raisebox{}[][]{}`, but the space between two letters is reduced through a negative value to `\hspace{}`. In the fourth example, on the other hand, not only the space between two letters is reduced, the vowels are printed in a smaller size. If a form like the ones shown in the third and fourth examples in Table 17.9 is to be used repeatedly, a shorter

Table 17.9 Texts at different levels and forms through the `\raisebox{}[][]{}` command

LaTeX input	**Output**
`Placement of two words may be seen here \raisebox{1ex}[3.5ex]{\bf Raised texts} raised with some vertical space above the line.`	Placement of two words may be seen here **Raised texts** raised with some vertical space above the line.
`Placement of two words may be seen here \raisebox{-1ex}[0ex][2.5ex]{\bf Lowered texts} lowered with some vertical space below the line.`	Placement of two words may be seen here **Lowered texts** lowered with some vertical space below the line.
`W\hspace{-0.2em}\raisebox{0.4ex}{A}\hspace{-0.2em}% V\hspace{-0.15em}\raisebox{0.4ex}{E}`	WAVE
`{\LARGE\bf~% D\hspace{-0.05em}\raisebox{0.55ex}{\large I}L% \hspace{-0.19em}\raisebox{0.55ex}{\large I}P~% D\hspace{-0.1em}\raisebox{0.15ex}{\large A}% \hspace{-0.15em}T\hspace{-0.13em}T% \hspace{-0.15em}\raisebox{0.15ex}{\large A}% }`	**DILIP DATTA**

new command may be defined (in the preamble) instead of using a long expression every time, e.g., ‘`\newcommand{\wave}{\mbox{W\hspace{-0.2em}\raisebox{0.4ex}{A} \hspace{-0.2em}V\hspace{-0.15em}\raisebox{0.4ex}{E}}}`’ to print WAVE using `\wave`. Note that the entire second argument of `\newcommand{}{}` is put here in `\mbox{}` in order to print WAVE without breaking or hyphening in between, as well as to make its effect local without affecting the remaining contents of a document.

17.4 Geometric Transformation of Items*

The **graphics** package has the provision for geometric transformations of texts and figures, such as scaling, rotation, and reflection. There are two commands for scaling, `\scalebox{}[]{}` and `\resizebox{}{}{}`. The `\scalebox{hsc}[vsc]{atext}` command scales `atext` in the horizontal direction by `hsc`, and also optionally in the vertical direction by `vsc` (`hsc` and `vsc` take numerical values only), while `\resizebox{hlen}{vlen}{atext}` prints `atext` in a horizontal length of `hlen` and a vertical height of `vlen` (`hlen` and `vlen` take values in units of length). The `\resizebox{hlen}{vlen}{atext}` command will print `atext` in proportion to `hlen` if the `!` symbol is used in place of `vlen`, while in proportion of `vlen` if `!` is used in place of `hlen`. On the other hand, the `\rotatebox{deg}{atext}` command rotates `atext` by an angle of `deg` in degree (a positive value of `deg` rotates `atext` in the counter-clockwise direction and a negative value in the clockwise direction), while the `\reflectbox{atext}` command reflects `atext` about a direction perpendicular to it.

Some geometric transformations made through the above four commands are shown in Table 17.10. Note that the commands can also be used in a combination for multiple transformations, like `\rotatebox{}{\scalebox{}[]{}}`, `\reflectbox{\rotatebox{}{}}`, and `\reflectbox{\rotatebox{}{\scalebox{}[]{}}}` as shown in examples 6–8 in Table 17.10.

Table 17.10 Geometric transformation of texts

#	LaTeX input	Output
1	`\scalebox{0.8}[2]{Scale} / \scalebox{2}[0.8]{Scale}`	Scale / Scale
2	`\resizebox{7mm}{6mm}{\bf Raise} / \resizebox{4cm}{4mm}{\bf Raise}`	**Raise / Raise**
3	`\resizebox{5mm}{!}{\bf Raise} / \resizebox{!}{5mm}{\bf Raise}`	**Raise / Raise**
4	`\rotatebox{30}{Rotate}`	Rotate
5	`\reflectbox{\bf Reflect}`	**Reflect**
6	`\rotatebox{20}{\scalebox{2}[0.8]{Rot.\␣scale}}`	Rot. scale
7	`\reflectbox{\rotatebox{-30}{\bf Refl.\␣rotate}}`	**Refl. rotate**
8	`\reflectbox{\rotatebox{-15}{%` `\scalebox{2}[0.8]{Refl.\␣rot.\␣scale}}}`	Refl. rot. scale
9	`{\LARGE\bf~%` `D\hspace{-0.07em}\raisebox{0.55ex}{\resizebox%` `{2.5mm}{3mm}{I}}\hspace{-0.05em}L\hspace%` `{-0.26em}\raisebox{0.55ex}{\resizebox{2.5mm}%` `{3mm}{I}}\hspace{-0.06em}P~D\hspace{-0.16em}%` `\raisebox{0.15ex}{\resizebox{6mm}{2.5mm}{A}}%` `\hspace{-0.29em}T\hspace{-0.13em}T\hspace%` `{-0.29em}\raisebox{0.15ex}{\resizebox{6mm}%` `{2.5mm}{A}}` `}`	**DILIP DATTA**

The last example in Table 17.10 is a more complex one. It combines the transformation command **\resizebox{}{}{}** with the raising command **\raisebox{}[][]{}** (refer §17.3 for detail) for printing characters in different levels and scales. Moreover, **\hspace{}** is used to adjust the space between two characters.

Hour 18

Miscellaneous II

Many special effects that can be produced in a document are discussed in Hour 17 on page 161. Some more effects are presented in this Hour, like hyperlinking a topic, verbatim texts, water-marking pages, inserting a logo, date, and time, etc.

18.1 Horizontal Rules and Dots

A horizontal rule (line) covering the entire width of a page, or a column in a multi-column document, can be drawn by the **\hrule** command. A shorter in-line horizontal rule, or rules of different widths at different heights, can be drawn by the **\rule[**`hgt`**]{**`hlen`**}{**`vlen`**}** command, where `hlen`, and `vlen` are, respectively, the horizontal and vertical lengths of the rule, while optional `hgt` is its height from the current line of texts. For example, **\rule{2cm}{1mm}** will draw ▬▬, while **\rule[2mm]{2cm}{1mm}** will draw ▬▬. As a special application, **\rule[ ]{}{}** may be used with zero width for raising or lowering an item, e.g., **\fbox{\rule[-2mm]{0mm}{6mm}**`Texts`**}** produces $\boxed{\text{Texts}}$ (while **\fbox{**`Texts`**}** produces $\boxed{\text{Texts}}$) by increasing the vertical height of **\fbox{}**.

There are many commands for producing different types of dots. The text-mode dot producing commands are **\dots** (…) and **\ldots** (…), while the math-mode commands include **\$\cdot\$** ($\cdot$), **\$\cdots\$** ($\cdots$), **\$\dotsb\$** ($\cdots$), **\$\dotsi\$** ($\cdots$), **\$\dotsm\$** ($\cdots$) and **\$\dotsc\$** ($\ldots$), where mainly the vertical positioning of the dots are noticeable.

18.2 Hyperlinking Referred and Cited Items

In the softcopy of a multi-page document, it is always preferred to have hyperlink to the referred and cited items, so that one can reach to those items just by a mouse click. The items which can be hyperlinked include page numbers in the lists of

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_18

contents and index, referred items (like sectional units, tables, figures, equations, theorems, etc.), cited references, and URLs. All of such hyperlinks can be obtained in LaTeX just by loading the **hyperref** package in the preamble of a document as **\usepackage[linktocpage=true]{hyperref}**, where the optional **linktocpage=true** instructs to hyperlink page numbers in the `Contents`, `List of Tables`, and `List of Figures`.

18.3 Current Date and Time*

The **\today** command produces current date in a standard format, like '`April 22, 2016`'. Other printing formats of **\today** can be obtained under the **datetime** package as follows:

- ▷ The **\today** command will print the current date like '`Friday 22`$^{\texttt{nd}}$ `April, 2016`' upon loading the **datetime** package without any option as **\usepackage{datetime}** or with the **dayofweek** option as **\usepackage[dayofweek]{datetime}**, or like '`22`$^{\texttt{nd}}$ `April, 2016`' with the **nodayofweek** option.
- ▷ The names of day and month are printed in full if **\today** is preceded by **\longdate** (default), i.e., simply **\today** or **\longdate\today**, while these are abbreviated to three alphabets (like `Fri` and `Apr`) if **\today** is preceded by **\shortdate** as **\shortdate\today**.
- ▷ For date in texts only, **\today** may be preceded by **\textdate** as **\textdate\today**, which will print '`Friday the Twenty-Second of April, Two Thousand and Sixteen`' under the **dayofweek** option to the **datetime** package, while '`Twenty-Second of April, Two Thousand and Sixteen`' under the **nodayofweek** option to the package.
- ▷ Date only in numerals can be obtained as '`22/04/2016`' with **\ddmmyyyydate\today**, '`22/4/2016`' with **\dmyyyydate\today**, '`22/04/16`' with **\ddmmyydate\today**, or '`22/4/16`' with **\dmyydate\today**. Date in numerals, but in the pattern of month-day-year, can be obtained if **\today** is preceded by such commands, such as **\mmddyyyydate**, **\mdyyyydate**, **\mmddyydate**, and **\mdyydate**. In the numerical date format, numbers can be separated by other symbols by redefining the **\dateseparator** command, e.g., **\renewcommand{\dateseparator}{-}** for separating two numbers by a hyphen instead of a slash, like '`22-04-2016`'.
- ▷ When the **datetime** package is active (i.e., loaded), date in the format like '`April 22, 2016`' can be obtained through **\usdate\today**.
- ▷ A user defined date format can be obtained through **\newdateformat{**`fname`**}{**`fuser`**}**, where `fuser` is the user defined format and `fname` is its name. In `fuser`, day can be defined by **\THEDAY**, **\twodigit{\THEDAY}** or **\ordinaldate{\THEDAY}**, while month can be defined by **\THEMONTH**, **\twodigit{\THEMONTH}**, **\monthname[\THEMONTH]** or **\shortmonthname[\THEMONTH]**. For example, defining a new date format as **\newdateformat{**`mydt`**}{\twodigit{\THEDAY}~\monthname[\THEMONTH],~\THEYEAR}**, date can be produced by '`\mydt`**\today**', which will print date as, e.g., '`22 April, 2016`'.

The **datetime** package also provides the **\currenttime** command for printing the current time. The format of time can be defined through the **\settimeformat{**`tformat`**}** command, where the permissible `tformat` are **xxivtime** (default), **ampmtime**, and **oclock**. Time will be printed like `16:59` with the **xxivtime** option, `4:59pm` with the **ampmtime** option, and '`One minute to Five in the afternoon`' with the **oclock** option.

18.4 Highlighted Texts*

It is discussed in §17.1 on page 161 how texts can be produced in various boxes, including colored boxes, for the purpose of making them prominent. Apart from that, the **soul** package defines the **\sethlcolor{}** and **\hl{}** commands, which can be used for highlighting some texts of a document by a specified color. First defining a color through **\sethlcolor{}**, the textual argument of **\hl{}** can be highlighted by that color. For example, the set of **\sethlcolor{**`ucgray`**}** and **\hl{**`This is highlighted by gray color`**}** commands will produce ' This is highlighted by gray color ', where `ucgray` is a gray color predefined through **\definecolor{**`ucgray`**}{gray}{**`0.75`**}**.

18.5 Verbatim Texts

As seen so far, the contents of a document are to be inserted in a LaTeX input file according to its fixed format. Many times it becomes difficult to format some texts, e.g., a computer program or simulated results of a program. LaTeX provides the **verbatim** environment, through which texts can be printed exactly the same way of their manual formatting. No LaTeX command or environment works in the **verbatim** environment, but it is simply printed as ordinary characters. Moreover, the special keyboard characters, given in Table 1.4 on page 8, can also be printed directly in the **verbatim** environment. An application of this environment is shown in Table 18.1,

Table 18.1 Manually formatted texts through the **verbatim** environment

LaTeX input	**Output**

\begin{verbatim}
`No \LaTeX command or any other environ-`
`ment works in the verbatim environment.`
`This is the only environment which`
`accepts manual formatting of a document.`
`Moreover, special keyboard characters,`
`such as $, %, ^, etc., can also be`
`printed directly through this`
`environment.`
\end{verbatim}
	`No \LaTeX command or any other environ-`
	`ment works in the verbatim environment.`
	`This is the only environment which`
	`accepts manual formatting of a document.`
	`Moreover, special keyboard characters,`
	`such as $, %, ^, etc., can also be`
	`printed directly through this`
	`environment.`

where it is seen that the **\LaTeX** command has become inactive in the **verbatim** environment (i.e., the '**\LaTeX**' command could not produce 'LaTeX', but it is printed just as ordinary characters). There is no provision for automatic line breaking also, but it is to be set manually by pressing the `Enter` button of the keyboard. Moreover, the special characters `$`, `%`, and `^` are also printed directly in the **verbatim** environment, which are to be printed in any other environment through the **\$**, **\%**, and **\^** commands, respectively.

The **verbatim** environment prints its contents in a new paragraph. The **\verb" "** command (or **\verb! !** replacing **"** with **!**) is used for printing verbatim texts in running texts, e.g., **\verb"**`\LaTeX`**"** or **\verb!**`\LaTeX`**!** prints '`\LaTeX`' in this line. Similarly, **\verb"**`a big      gap`**"** will print '`a big      gap`'. There also exist **\verb*" "** and **\verb*! !** commands, which print a ␣ in each blank space, e.g., **\verb*"**`a big      gap`**"** will print '`a␣big␣␣␣␣␣␣gap`'. The **verbatim** environment is generally used for large texts such as a paragraph, while the **\verb" "** and **\verb*" "** (or **\verb! !** and **\verb*! !**) commands are used for short inline texts such as one or two words[1]. The **verbatim** environment and the **\verb" "** and **\verb*" "** commands may not work as the arguments of other commands. However, they can be used in another environment.

18.5.1 Boxed and Listed Verbatim Texts

Like the **boxedminipage** environment (refer §4.4 on page 31 for detail), the **boxedverbatim** environment defined in the **moreverb** package may be used for printing verbatim texts in a box. However, unlike the **boxedminipage** environment, the **boxedverbatim** environment does not take any alignment or size argument, i.e., its simple structure is **\begin{boxedverbatim}**... **\end{boxedverbatim}**. Its effect can be seen by replacing **verbatim** in Table 18.1 with **boxedverbatim**.

The **moreverb** package provides the **listing** environment also, which numbers its contents starting as **\begin{listing}[**`astep`**]{**`n`**}**, where mandatory `n` is the starting line number and optional `astep` is the step size for numbering subsequent lines. Table 18.2 on the next page shows two applications of the **listing** environment. Since numbering is started with 1 without any option for step size, the lines in the first example in Table 18.2 are numbered serially starting from 1. On the other hand, numbering in the second example is started with 52 and thereafter only alternate lines are numbered because of the optional step size of 2.

18.5.2 Verbatim Texts with LaTeX Commands*

The verbatim texts producing commands and environments (**\verb" "**, **\verb*" "**, **\verb! !**, and **\verb*! !** commands, and **verbatim**, **boxedverbatim**, and **listing** environments discussed above) print everything, entered from a keyboard, blindly as ordinary characters. Therefore, these cannot be used for producing symbols or mathematical expressions, which are usually not available in a keyboard but are to be generated

[1]The **verbatim** environment is used for large texts such as a paragraph, while the **\verb" "** and **\verb*" "** (or **\verb! !** and **\verb*! !**) commands are used for short inline texts like one or two words.

Table 18.2 Line numbering of verbatim texts through the **listing** environment under the **moreverb** package

LaTeX input	Output

\begin{listing}{1}
for(i = 1; i <= n-1; i++)
{   for(j = i+1; j <= n; j++)
    {   if(a[i] < a[j])
        {   tmp = a[i]
            a[i] = a[j]
            a[j] = tmp
        }
    }
}
\end{listing}
	1 for(i = 1; i <= n-1; i++)
	2 { for(j = i+1; j <= n; j++)
	3 { if(a[i] < a[j])
	4 { tmp = a[i]
	5 a[i] = a[j]
	6 a[j] = tmp
	7 }
	8 }
	9 }

\begin{listing}[2]{52}
for(i = 1; i <= n-1; i++)
{   for(j = i+1; j <= n; j++)
    {   if(a[i] < a[j])
        {   tmp = a[i]
            a[i] = a[j]
            a[j] = tmp
        }
    }
}
\end{listing}
	52 for(i = 1; i <= n-1; i++)
	        { for(j = i+1; j <= n; j++)
	54 { if(a[i] < a[j])
	                { tmp = a[i]
	56 a[i] = a[j]
	                    a[j] = tmp
	58 }
	            }
	60 }

through some LaTeX syntax. Such difficulties can be sorted out in the **alltt** environment defined in the **alltt** package.

The **alltt** environment also acts like the **verbatim** environment, except that a backslash (****) and the curly braces (**{ }**) retain their usual LaTeX modes, which allow other commands and environments to work in the **alltt** environment[2]. Moreover, the **alltt** environment performs automatic line breaking also. Table 18.3 shows an example of

Table 18.3 Preserving LaTeX syntax in verbatim texts through the **alltt** environment

LaTeX input	Output
\begin{alltt} The {\it alltt} environment can be used to print verbatim texts preserving other \LaTeX\ commands and environments, as well as with automatic line breaking. Therefore, \(\beta\sb{1}=y\sp{2}\int{f(x)\,dx}\) in the {\it alltt} environment will produce ... \end{alltt}	The *alltt* environment can be used to print verbatim texts preserving other LaTeX commands and environments, as well as with automatic line breaking. Therefore, \(\beta\sb{1}=y\sp{2}\int{f(x)\,dx}\) in the *alltt* environment will produce $\beta_1 = y^2 \int f(x)\,dx$.

the **alltt** environment, where the syntax '**\LaTeX**' in the input file prints the word 'LaTeX' in the output, unlike `\LaTeX` in the **verbatim** environment as shown in Table 18.1. Moreover, a mathematical expression could also be produced through

[2]The backslash (****) and curly braces (**{ }**) retain their usual LaTeX modes in the verbatim texts generating **alltt** environment, which allow other commands and environments to work in this environment.

the inline math-mode \(. . .\) (mathematical environments, like equation or eqnarray, do not work in the alltt environment). On the other hand, the _ and ˆ symbols, usually used in math-mode for producing subscript and superscript, respectively, act as ordinary characters in the alltt environment, even inside the inline math-mode \(. . .\). Therefore, the \sb{} and \sp{} commands are used in the alltt environment in Table 18.3 for producing a subscript and a superscript, respectively.

18.6 Fragile Commands

Many LaTeX commands are fragile, such as \begin{}, \centering, or \footnote{}. All commands having optional arguments, as well as almost all starred-form commands, are fragile[3]. Fragile commands are not directly acceptable in the arguments of many other commands, like sectional commands of \chapter{} or \section{}. In such cases, a fragile command is to be used in protected mode through the \protect command. Such an example is shown in Table 18.4, where the argument of the \section{} command is center-aligned through the protected \centering command as \protect\centering. Moreover, a foot note is also generated in the \section{} command through the protected \footnote{} command as \protect\footnote{}. Note that each fragile command is to be protected by a separate \protect command.

Table 18.4 Fragile commands in protected mode

LaTeX input	**Output**

`\section{\protect\centering Protected fragile commands\protect\footnote{A fragile command is defined...}}`
`%`
`Fragile commands can be protected in the arguments of other commands ...`
	3.1 Protected fragile commands[1]
	Fragile commands can be protected in the arguments of other commands ...
	
	[1]A fragile command is defined . . .

[3]All commands having optional arguments as well as almost all starred-form commands are fragile, which are to be protected through \protect if inserted in the argument of other commands.

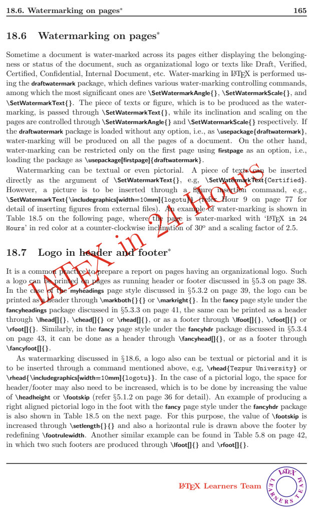
18.7 Watermarking on Pages*

Sometime a document is water-marked across its pages either displaying the belongingness or status of the document, such as organizational logo or texts like Draft, Verified, Certified, Confidential, Internal Document, etc. Water-marking in LaTeX is performed using the **draftwatermark** package, which defines various water-marking controlling commands, among which the most significant ones are **\SetWatermarkAngle{}**, **\SetWatermarkScale{}**, and **\SetWatermarkText{}**. The piece of texts or figure, which is to be produced as the water-marking, is passed through **\SetWatermarkText{}**, while its inclination and scaling on the pages are controlled through **\SetWatermarkAngle{}** and **\SetWatermarkScale{}**, respectively. If the **draftwatermark** package is loaded without any option, i.e., as **\usepackage{draftwatermark}**, water-marking will be produced on all the pages of a document. On the other hand, water-marking can be restricted only on the first page using **firstpage** as an option, i.e., loading the package as **\usepackage[firstpage]{draftwatermark}**.

Watermarking can be textual or even pictorial. A piece of texts can be inserted directly as the argument of **\SetWatermarkText{}**, e.g., **\SetWatermarkText{**`Certified`**}**. However, a picture is to be inserted through a figure insertion command, e.g., **\SetWatermarkText{\includegraphics[width=**`10`**mm]{**`logotu`**}}** (refer Hour 9 on page 81 for detail of inserting figures from external files). An example of water-marking is shown in Table 18.5, where the page is water-marked with 'LaTeX `in 24 Hours`' in red color at a counter-clockwise inclination of 30° and a scaling factor of 2.5.

Table 18.5 Watermarking and logo on pages

LaTeX input	Output

`\documentclass{report}`
`...`
`% Following 5 lines for watermarking`
`\usepackage{draftwatermark}`
`\SetWatermarkAngle{30}`
`\SetWatermarkScale{2.5}`
`\SetWatermarkText{\textcolor{red}{%`
`                 \LaTeX\␣in 24 Hours}}`
`% Following 6 lines for logo as footer`
`\usepackage{fancyhdr}`
`\pagestyle{fancy}`
`\renewcommand{\footrulewidth}{0.3pt}`
`\setlength{\footskip}{2.5cm}`
`\fancyfoot[R]{\includegraphics%`
`                 [height=2.0cm]{logo}}`
`...`
`\begin{document}`
`...`
`\section{Watermarking on pages*}`
`Sometime a document is water-marked`
`across its pages ...`
`...`
`\end{document}`
	

18.6. Watermarking on pages* 165

18.6 Watermarking on pages*

Sometime a document is water-marked across its pages either displaying the belongingness or status of the document, such as organizational logo or texts like Draft, Verified, Certified, Confidential, Internal Document, etc. Water-marking in LaTeX is performed using the **draftwatermark** package, which defines various water-marking controlling commands, among which the most significant ones are **\SetWatermarkAngle{}**, **\SetWatermarkScale{}**, and **\SetWatermarkText{}**. The piece of texts or figure, which is to be produced as the water-marking, is passed through **\SetWatermarkText{}**, while its inclination and scaling on the pages are controlled through **\SetWatermarkAngle{}** and **\SetWatermarkScale{}** respectively. If the **draftwatermark** package is loaded without any option, i.e., as **\usepackage{draftwatermark}**, water-marking will be produced on all the pages of a document. On the other hand, water-marking can be restricted only on the first page using **firstpage** as an option, i.e., loading the package as **\usepackage[firstpage]{draftwatermark}**.

Watermarking can be textual or even pictorial. A piece of texts can be inserted directly as the argument of **\SetWatermarkText{}**, e.g, **\SetWatermarkText{**`Certified`**}**. However, a picture is to be inserted through a figure insertion command, e.g., **\SetWatermarkText{\includegraphics[width=**`10`**mm]{**`logotu`**}}** (refer Hour 9 on page 77 for detail of inserting figures from external files). An example of water-marking is shown in Table 18.5 on the following page, where the page is water-marked with 'LaTeX `in 24 Hours`' in red color at a counter-clockwise inclination of 30° and a scaling factor of 2.5.

18.7 Logo in header and footer*

It is a common practice to prepare a report on pages having an organizational logo. Such a logo can be printed on pages as running header or footer discussed in §5.3 on page 38. In the case of the **myheadings** page style discussed in §5.3.2 on page 39, the logo can be printed as a header through **\markboth{}{}** or **\markright{}**. In the **fancy** page style under the **fancyheadings** package discussed in §5.3.3 on page 41, the same can be printed as a header through **\lhead[]{}**, **\chead[]{}** or **\rhead[]{}**, or as a footer through **\lfoot[]{}**, **\cfoot[]{}** or **\rfoot[]{}**. Similarly, in the **fancy** page style under the **fancyhdr** package discussed in §5.3.4 on page 43, it can be done as a header through **\fancyhead[]{}**, or as a footer through **\fancyfoot[]{}**.

As watermarking discussed in §18.6, a logo also can be textual or pictorial and it is to be inserted through a command mentioned above, e.g, **\rhead{**`Tezpur University`**}** or **\rhead{\includegraphics[width=**`10`**mm]{**`logotu`**}}**. In the case of a pictorial logo, the space for header/footer may also need to be increased, which is to be done by increasing the value of **\headheight** or **\footskip** (refer §5.1.2 on page 36 for detail). An example of producing a right aligned pictorial logo in the foot with the **fancy** page style under the **fancyhdr** package is also shown in Table 18.5 on the next page. For this purpose, the value of **\footskip** is increased through **\setlength{}{}** and also a horizontal rule is drawn above the footer by redefining **\footrulewidth**. Another similar example can be found in Table 5.8 on page 42, in which two such footers are produced through **\lfoot[]{}** and **\rfoot[]{}**.

18.8 Logo in Header and Footer*

It is a common practice to prepare a report on pages having an organizational logo. Such a logo can be printed on pages as running header or footer discussed in §5.3 on page 40. In the case of the **myheadings** page style discussed in §5.3.2 on page 42, the logo can be printed as a header through **\markboth{}{}** or **\markright{}**. In the **fancy** page style under the **fancyheadings** package discussed in §5.3.3 on page 43, the same can be printed as a header through **\lhead[]{}**, **\chead[]{}** or **\rhead[]{}**, or as a footer through **\lfoot[]{}**, **\cfoot[]{}** or **\rfoot[]{}**. Similarly, in the **fancy** page style under the **fancyhdr** package discussed in §5.3.4 on page 45, it can be done as a header through **\fancyhead[]{}**, or as a footer through **\fancyfoot[]{}**.

As watermarking discussed in §18.7 on page 177, a logo also can be textual or pictorial and it is to be inserted through a command mentioned above, e.g., **\rhead{**`Tezpur University`**}** or **\rhead{\includegraphics[width=**`10`**mm]{**`logotu`**}}**. In the case of a pictorial logo, the space for header/footer may also need to be increased, which is to be done by increasing the value of **\headheight** or **\footskip** (refer §5.1.2 on page 38 for detail). An example of producing a right aligned pictorial logo in the foot with the **fancy** page style under the **fancyhdr** package is also shown in Table 18.5. For this purpose, the value of **\footskip** is increased through **\setlength{}{}** and also a horizontal rule is drawn above the footer by redefining **\footrulewidth**. Another similar example can be found in Table 5.8, in which two such footers are produced through **\lfoot[]{}** and **\rfoot[]{}**.

18.9 Paragraphs in Different Forms*

The **picinpar** package provides the facility for creating a window within a paragraph, in which some other texts, tables, and figures can be printed. These are done through the **window**, **tabwindow**, and **figwindow** environments, respectively. Each of these environments takes four mandatory fields in **[]** separating two by a comma, and then the paragraph in the environment. Accordingly, the structure of an environment becomes **\begin{**`env`**}[**`nline`**,**`halgn`**,**`\wcmd`**{**`wmat`**},**`wnote`**]**`apara`**\end{**`env`**}**, where `env` is the name of the environment, `apara` is the paragraph, `nline` is the number of lines of `apara` after which the window is to be created, `halgn` is the horizontal alignment of the window in `apara` (**l** for left alignment, **c** for centered and **r** for right alignment), `\wcmd`**{}** is the window generating command, `wmat` is the material to be produced through `\wcmd`**{}**, and `wnote` is a note about `wmat` (such as the caption of a table or a figure). Applications of these three environments are shown in Table 18.6 on the following page. The **\shortstack[]{}** command, used in the first example in Table 18.6, vertically stacks the lines of its mandatory argument one below another (the optional argument of **\shortstack[]{}** is for horizontal positioning).

Table 18.6 A window within a paragraph

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><code>\begin{window}[1,c,%
    \fbox{\shortstack{U\\R\\G\\E\\N\\T}},{ }]
The window, tabwindow and figwindow
environments allow to create windows inside
paragraphs for inserting other texts,
tables, and figures, respectively ...
\end{window}</code></td><td>The window, tabwindow and figwindow environments allow to create windows inside paragraphs for inserting other texts, tables, and figures, respectively. Each of these environments takes four mandatory fields in [] separating two by a comma, and then the paragraph in the environment.
U
R
G
E
N
T</td></tr>
<tr><td><code>\begin{tabwindow}[1,r,%
    \fbox{\begin{tabular}{l|l}%
    Rice&20.00\\Oil&60.00\\Wheat&25.00%
    \end{tabular}},{Prices}]
See the price list. Prices of items are
increasing everyday. It has become difficult
for low-income people to survive ...
\end{tabwindow}</code></td><td>See the price list. Prices of items are increasing everyday. It has become difficult for low-income people to survive. Along with prices, daily needs are also increasing, but there is no increase in income for quite a long time. Who knows when the Government will take some steps to overcome the situation ...
<table><tr><td>Rice</td><td>20.00</td></tr><tr><td>Oil</td><td>60.00</td></tr><tr><td>Wheat</td><td>25.00</td></tr></table>Table 3: Prices.</td></tr>
<tr><td><code>\begin{figwindow}[2,r,%
    \epsfig{file=fwork.eps},{Finger work.}]
Tanushree is over smart at her age of 3
years only. She is simply a copycat of her
elder sister Devoshree ...
\end{figwindow}</code></td><td>Tanushree is over smart at her age of 3 years only. She is simply a copycat of her elder sister Devoshree. Can do or not, but she follows everything that the elder sister performs. The picture on the right side shows her finger work, which is decorated by the elder sister. Tanushree hardly cries, but remains smiling. Wish both the sisters will always remain friendly ...

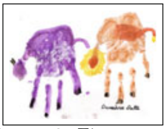



Figure 6: Finger work.</td></tr>
</table>

There is another package, **shapepar**, which defines the **\squarepar{}**, **\diamondpar{}**, **\heartpar{}**, and **\shapepar\nutshape{}** commands for printing a textual paragraph (the argument of a command) in the shape of a square, diamond, heart, and nut (hexagonal outer shape and circular inner shape), respectively. Applications of these commands are shown in Table 18.7 on the next page for the same textual argument. It is really interesting to see that the paragraph under the **\diamond{}** command is started and ended with diamond symbol, while the paragraph under the **\heart{}** command is ended with a heart symbol. Note that the **\shapepar\nutshape{}** command is to be followed by a new paragraph (i.e., a blank line, **\par** command, or other sectional command like **\section{}** or **\subsection{}**), otherwise the contents following the command will also be printed in a nut shape merging with its argument.

Table 18.7 Paragraphs of different shapes under the **shapepar** package

LaTeX input	`\squarepar{This command may be used if a textual argument is to be printed in the shape of a square.}`	`\diamondpar{This command may be used if a textual argument is to be printed in the shape of a diamond.}`	`\heartpar{This command may be used if a textual argument is to be printed in the shape of a heart.}`	`\shapepar\nutshape{ This command may be used if a textual argument is to be printed in the shape of a nut.}`
Output	This command may be used if a textual argument is to be printed in the shape of a square.	◇ This command may be used if a textual argument is to be printed in the shape of a diamond. ◇	This command may be used if a textual argument is to be printed in the shape of a heart. ♡	This command may be used if a textual argument is to be printed in the shape of a nut.

Hour 19

Letter and Article

LaTeX based procedures for producing different components of a document are discussed in previous Hours. A full document can be prepared by using those procedures. There are several standard formats for producing different types of documents, such as letter, article, report, and book. As a quick recap, a LaTeX input file is started by **\documentclass[ ]{ }** with the mandatory argument in **{ }** as the class of the document, like **letter**, **article**, **report** or **book**. Options, if any, are inserted in **[ ]** separating two options by a comma, such as '**a4paper,**12**pt**' for producing a document on A4-size paper in 12 point fonts. After **\documentclass[ ]{ }**, various packages and other global declarations are inserted in the preamble. Finally, the contents of the document to be produced are inserted in the body of the input file, i.e., in the **document** environment (refer §1.3 on page 2 for detail of the general format of a LaTeX input file).

19.1 Letter Writing

A letter is prepared through the document-class of **letter**, which contains some standard commands for producing different parts of a letter. Such commands are shown in Table 19.1 in the order of their application (these are not mandatory commands,

Table 19.1 Standard commands under the document-class **letter**

LaTeX command	Function
\address{`Sender`**}**	`Sender`: Sender's address at the top-right corner.
\signature{`Signature`**}**	`Signature`: Sender's signature (name) at the bottom-centre.
\begin{letter}{`Recipient`**}**	`Recipient`: Recipient's address on the left side.
\opening{`Salute`**}**	`Salute`: Addressing the recipient before starting the contents.
\closing{`Anticipate`**}**	`Anticipate`: Anticipating the recipient at the end of the contents.
\cc{`Copy`**}**	`Copy`: List of persons whom to send a copy of the letter.
\encl{`Enclosure`**}**	`Enclosure`: List of enclosures with the letter.

but can be used as per requirement). The **\address{ }** and **\signature{ }** commands are inserted in the **document** environment prior to starting the **letter** environment, while

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_19

\opening{}, \closing{}, \cc{} and \encl{} are inserted in the letter environment. The actual contents of the letter are inserted between \opening{} and \closing{}.

The general format of a letter in terms of the commands of Table 19.1 is shown in Table 19.2. It is seen in the output that by default the current date in a specified

Table 19.2 Standard format of the document-class letter

LaTeX input	Output

`% File Name: myletter.tex`
`\documentclass[a4paper,12pt]{letter}`
`\begin{document}`
`\address{Sender's Address}`
`\signature{Sender's Name}`
`\begin{letter}{Recipient's Address}`
`\opening{Dear Sir,}`
`Contents of the letter ...`
`\closing{Best regards,}`
`\cc{1. Secretary\\2. Coordinator}`
`\encl{1. Letter from CEO.\\2. Letter from MD.}`
`\end{letter}`
`\end{document}`
	Sender's Address
	June 10, 2017
	Recipient's Address
	Dear Sir,
	Contents of the letter ...
	Best regards,
	Sender's Name
	cc: 1. Secretary
	2. Coordinator
	encl: 1. Letter from CEO.
	2. Letter from MD.

format is also printed below the sender's address. The default format of the date can be changed through the \date{} command (in the preamble) with the required format as its argument, e.g., \date{29/02/2016} for printing 29/02/2016. On the other hand, if the date is not required to be printed, simply the \date{} command with empty argument may be used.

The standard formatting of the document-class letter, shown in Table 19.2, may not be suitable in some cases. Rather, one may prefer to prepare a letter on his/her own style. Table 19.3 on the next page shows such a letter without using any standard formatting command. The setspace package is used for setting line spacing through the spacing environment, as done in Table 19.3 using \begin{spacing}{1.2}. The \pagestyle{empty} command prevents the page numbering of the letter, and the \today command prints the date of compilation of the letter (refer §18.3 on page 172 for detail). The \hspace*{\fill} command is used at the starting of three lines for right aligning the texts of those lines.

Note that the letter prepared in Table 19.2 using the standard format of the document-class letter is assigned the compilation date, but not page numbering. In contrary, the letter in a user-specified format in Table 19.3 is assigned page numbering (which is prevented through the \pagestyle{empty} command), but not the compilation date.

19.2 Article Preparation

The templates of articles for publishing in journals, proceedings, magazines, etc., vary from publisher to publisher. Many publishers provide their own templates for maintaining uniformity in a volume, and an author needs just to insert the contents of

Table 19.3 A user-specified format in the document-class **letter**

<table>
<tr><th>LaTeX input</th><th>Output</th></tr>
<tr><td><pre>
\documentclass[a4paper,12pt]{letter}
\usepackage{setspace}
\pagestyle{empty}
%
\begin{document}
From:\\Sender's Address\\[2mm]
To:\\Recipient's Address\\
\hspace*{\fill} Date: \today\\[2mm]
{\bf Subject: Regarding ...}\\
%
\begin{spacing}{1.2}
Respected Sir,\\
This is to inform you that ...
\par
Therefore, hereby I request you to ...
\vskip 5mm
\hspace*{\fill}Thanking you,\\[7mm]
\hspace*{\fill}(Sender's Name)\\[2mm]
{\it Copy to\/}: President\\[2mm]
{\bf Enclosure:} Detail of the findings.
\end{spacing}
\end{document}
</pre></td><td>From:
Sender's Address

To:
Recipient's Address
Date: June 10, 2017

Subject: Regarding ...

Respected Sir,
This is to inform you that ...
Therefore, hereby I request you to ...

Thanking you,

(Sender's Name)

Copy to: President

Enclosure: Detail of the findings.</td></tr>
</table>

an article in the given template. If not given, authors can prepare articles in their own templates. A number of such LaTeX based standard templates are discussed here.

An Article can be prepared in the document-class of **article** or **amsart**. Generally an article starts with a title[1] and the list of author(s), which are inserted as the arguments of the **\title{}** and **\author{}** commands respectively. The line break command (****), if required, is permissible in the arguments of these two commands. The **\title{}** and **\author{}** commands are activated using the **\maketitle** command in the **document** environment before inserting any content of the article. The **\title{}** and **\author{}** commands can be used either in the preamble or even in the **document** environment, but must be before the **\maketitle** command. If used, the optional command **\date{}** goes along with the **\title{}** and **\author{}** commands. Following the **\maketitle** command, the abstract of the article is inserted in the **abstract** environment[2]. Then the actual contents of the article are inserted through a series of standard formatting, such as **\section{}**, **\subsection{}**, **\subsubsection{}**, **\paragraph{}**, and **\subparagraph{}** commands, as well as other applicable commands and environments discussed up to the previous Hour.

The general format of an article, in both the document-classes of **article** and **amsart**, are shown in Tables 19.4 and 19.5 on the next page, where the differences in the outputs of the two document-classes are clearly visible. The document-class **article** by default prints the compilation date of the article (which is prevented in Table 19.4 through the **\date{}** command with empty argument). The heading of each section and subsection is full-aligned, and its contents are started from a new line. In contrary,

---

[1]The **titlepage** option in the **\documentclass[]{}** command produces the title of a document on a separate page.

[2]The **abstract** environment works in the document-classes of **article** and **amsart**, but not in **book**.

Table 19.4 Article in the document-class **article**

LaTeX input	Output
% myarticle.tex (in 'article') \documentclass[a4paper,12pt]{article} \date{} \title{My First Article in \LaTeX} \author{Author's Name and Address} \begin{document} \maketitle % \begin{abstract} The article explains ... \end{abstract} % \section{First Section} First level of numbered section. \subsection{First subsection} Second level of numbered section. \subsubsection{First sub-subsection} Third and last level of numbered section. \section{Second Section} Texts of the second section ... % \end{document}	My First Article in LaTeX Author's Name and Address **Abstract** The article explains ... **1 First Section** First level of numbered section. **1.1 First subsection** Second level of numbered section. **1.1.1 First sub-subsection** Third and last level of numbered section. **2 Second Section** Texts of the second section ... 1

Table 19.5 Article in the document-class **amsart**

LaTeX input	Output
% myarticle.tex (in 'amsart') \documentclass[a4paper,12pt]{amsart} \title{My First Article in \LaTeX} \author{Author's Name and Address} \begin{document} \maketitle % \begin{abstract} The article explains ... \end{abstract} % \section{First Section} First level of numbered section. \subsection{First subsection} Second level of numbered section. \subsubsection{First sub-subsection} Third and last level of numbered section. \section{Second Section} Texts of the second section ... % \end{document}	**MY FIRST ARTICLE IN LATEX** AUTHOR'S NAME AND ADDRESS ABSTRACT. The article explains ... 1. FIRST SECTION First level of numbered section. 1.1. **First subsection.** Second level of numbered section. 1.1.1. *First sub-subsection.* Third and last level of numbered section. 2. SECOND SECTION Texts of the second section ... 1

the heading of a section in the document-class **amsart** is center-aligned. Moreover, the contents of the abstract and each subsection are printed in continuation of their headings. On the other hand, the title, author's name and address, and the headings of abstract and sections are always printed in uppercase letters, even if these are inserted in lowercase letters in the LaTeX input file.

The default heading of the **abstract** environment in the document-classes **article** and **amsart** can be changed by redefining the **\abstractname** command in the preamble, e.g., `\renewcommand{\abstractname}{Summary}` for replacing the heading ‘Abstract’ by ‘Summary’.

19.2.1 List of Authors

Tables 19.4 and 19.5 show only one author in the article, which is center-aligned. When the number of authors is more than one, these may be printed one below another, side-by-side, or in any other user-defined format. A format for two authors, printed one below another, is shown Table 19.6.

Table 19.6 Authors in articles one below another

LaTeX input	Output

`\author`
`{`
`    {\bf 1st author’s name}\\`
`    Affiliation\\`
`    Address\\[2mm]`
`    %`
`    {\bf 2nd author’s name}\\`
`    Affiliation\\`
`    Address`
`}`
	1st author’s name
	Affiliation
	Address
	
	2nd author’s name
	Affiliation
	Address

Another format is shown in Table 19.7, where three authors are printed

Table 19.7 Authors side-by-side through the **tabular** environment

```
\author
{
    \begin{tabular}[t]{c@{\extracolsep{30mm}}c@{\extracolsep{30mm}}c}
     {\it Author-1}   &  {\it Author-2}   &  {\it Author-3}\\
     Affiliation      &  Affiliation      &  Affiliation\\
     Address          &  Address          &  Address\\
     e-mail           &  e-mail           &  e-mail\\
    \end{tabular}
}
```

Output		
Author-1	*Author-2*	*Author-3*
Affiliation	Affiliation	Affiliation
Address	Address	Address
e-mail	email	email

side-by-side through the **tabular** environment with three columns. The contents of each column are center-aligned and two columns are separated by extra 30 mm space through the `@{\extracolsep{30mm}}` command.

A third format is shown in Table 19.8, where the detail of an author is printed

Table 19.8 Author details at the bottom of a page through the \thanks{} command

LaTeX input	Output

\documentclass[a4paper,12pt]{article}
\date{}
\title{My First Article in \LaTeX}
\author
{
Mr.\,X\thanks{X's Address} \and
Mr.\,Y\thanks{Y's Address}
}
\begin{document}
\maketitle
%
\begin{abstract}
The article explains ...
\end{abstract}
%
\section{Introduction}
Introduction to the problem ...
%
\end{document}
	My First Article in LaTeX
	Mr. X* Mr. Y†
	Abstract
	The article explains ...
	1 Introduction
	Introduction to the problem ...
	*X's Address
	†Y's Address
	1

at the bottom of the page. This is done through the \thanks{} command after the name of each author, where the detail of an author is inserted as the argument of the command. The same effect can be obtained through the \footnote{} command also, instead of the \thanks{} command. The \and command is used between the names of the two authors to separate them by a big gap in the output.

19.2.2 Title and Abstract on Separate Pages

Some publishers may ask to produce title, list of authors and abstract of an article on separate pages, particularly for the review purpose. These can be achieved by using the titlepage and abstract options in the \documentclass[ ]{} command, i.e., as \documentclass[titlepage,abstract]{article}. The titlepage option instructs \maketitle to produce the title and list of authors on a separate page, while the abstract option instructs to produce the abstract on another separate page.

*19.2.3 Left Aligned Title and List of Authors**

Notice in §19.2.1 that the title and list of authors of an article are always center aligned. Sometime these may need to be left aligned, which can be achieved simply by inserting the following few lines of commands in the preamble:

```
\makeatletter
\def\maketitle
{   {\bf\Large\raggedright \@title} \vskip 5mm
    {\large\raggedright \@author} \vskip 10mm
}
\makeatother
```

where the pair of \makeatletter and \makeatother commands brackets a command, starting with a '@' (commands starting with a '@' are LaTeX's internal commands), to work as an ordinary command. The \raggedright command here makes \@title (title) and \@author (list of authors) left aligned. The \bf, \Large and \large are just text formatting commands, while the \vskip command is used for creating some vertical blank space after the title and the list of authors.

19.2.4 Articles in Multiple Columns

Many publishers want articles to be produced in multiple columns. The **twocolumn** option in the \documentclass[ ]{ } command produces an article in two columns. In the document-class **article**, as shown in Table 19.9, the title and list of authors (i.e., the

Table 19.9 Article in two columns through the **twocolumn** option in \documentclass[ ]{ }

LaTeX input	\documentclass[a4paper,12pt,twocolumn]{article} \date{} \title{My First Article in \LaTeX} \author{Author's Name and Address} \begin{document} \maketitle % \begin{abstract} Abstract of the article … Abstract of the article … \end{abstract} % \section{Introduction} Introduction to the work … Introduction to the work … \end{document}
Output	My First Article in LaTeX Author's Name and Address **Abstract** Abstract of the article … Abstract of the article … Abstract of the article … Abstract of the article … **1 Introduction** Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work …

arguments of \title{} and \author{}) are printed in single-column, and the abstract and other contents of the article are printed in two columns. In the case of the document-class amsart, however, the title and list of authors are also printed in two columns.

Sometime the title, author and abstract may need to be printed in a single column, while the rest of the article in two columns. In that case, instead of the twocolumn option in \documentclass[]{}, the \twocolumn[] command may be used[3]. As shown in Table 19.10, \twocolumn[] is used after \begin{document}, putting the \maketitle command and the abstract environment in [] of \twocolumn[] for printing them in a single column. After \end{abstract}, the \vspace{1.0cm} command is used to leave 1.0 cm vertical blank space before starting the two-column mode.

Table 19.10 Article in two columns through the \twocolumn[] command

LaTeX input	\documentclass[a4paper,12pt]{article}
	\date{}
	\title{My First Article in \LaTeX}
	\author{Author's Name and Address}
	\begin{document}
	\twocolumn
	[
	    \maketitle
	    \begin{abstract}
	    Abstract of the article … Abstract of the article …
	    \end{abstract}
	    \vspace{1.0cm}
	]
	\section{Introduction}
	Introduction to the work … Introduction to the work …
	\end{document}

Output	My First Article in LaTeX Author's Name and Address **Abstract** Abstract of the article … Abstract of the article … Abstract of the article … Abstract of the article … Abstract of the article … Abstract of the article … **1 Introduction** Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work … Introduction to the work …

Note that the \onecolumn[] and \twocolumn[] commands can be used alternatively for producing different parts of a document alternatively in single column and two columns, respectively. However, each of such parts will be produced on a new page even if sufficient blank space is available on the previous page. Therefore, instead

[3] The twocolumn option may be used in \documentclass[]{} if an entire article is to be produced in two columns, while \twocolumn[] may be used if some components, like title, author and abstract, are to be produced in a single column and the rest in two columns.

of using the **\onecolumn[]** and **\twocolumn[]** commands alternatively, the process discussed in §4.3.2 on page 30 may be followed for such requirement.

*19.2.5 Section-Wise Numbering of Items**

Numbered items (like tables, figures, and equations) in the document-classes **report** and **book** are numbered chapter-wise, i.e., the numbering style is composed of two parts – the serial number of the chapter and the serial number of the item, separated by a period (as can be seen in this book). In contrast, in the document-class **article**, these items are numbered by their serial numbers only, i.e., not preceded by the serial number of the section in which the items belong (the document-class **article** does not support a chapter). If these items are to be numbered section-wise in the document-class **article**, the following few lines of commands may be included in the preamble:

```
\makeatletter
\@addtoreset{table}{section}
\@addtoreset{figure}{section}
\@addtoreset{equation}{section}
\makeatother
\renewcommand{\thetable}{\thesection.\arabic{table}}
\renewcommand{\thefigure}{\thesection.\arabic{figure}}
\renewcommand{\theequation}{\thesection.\arabic{equation}}
```

where **\@addtoreset{}{}** resets its first argument according to the second argument, i.e., to number tables, figures, and equations section-wise when the above codings are applied. By redefining **\thetable**, **\thefigure** and **\theequation** through **\renewcommand{}{}**, the default numbering styles of tables, figures and equations are altered to those specified as the second argument of **\renewcommand{}{}**. According to the above coding, each numbering will start with the serial number of the section, followed by a period and then the serial number of the item in an Arabic numeral (due to **\thesection**, '**.**' mark, and **\arabic{}**, respectively). Without **\renewcommand{}{}** in the above coding, although the items will be numbered section-wise internally, the section numbers will not be visible in the output. Such numbering would be confusing to understand, particularly when they will be referred somewhere, as the same numbering, like `Figure 1` or `Figure 2`, will be repeated in every section. An application of the above coding is shown in §11.4.3 on page 105.

*19.2.6 Dividing an Article into Parts**

Sections of an article can be divided into parts through **\part{}**. Each **\part{}** generates the label-word '`Part`' followed by a serial number in a uppercase Roman numeral (such as `Part I` or `Part II`), and then prints its argument as the heading of the part.

Although divided into a number of parts, by default the sections of an article will still be assigned continuous serial numbers irrespective of the parts in which they belong. To number the sections part-wise, the following set of commands may be used in the preamble:

```
\makeatletter
\@addtoreset{section}{part}
\makeatother
\renewcommand{\thesection}{\thepart.\arabic{section}}
```

where \@addtoreset{section}{part} resets the sections to be numbered part-wise. The \renewcommand{\thesection}{\thepart.\arabic{section}} command redefines the numbering of sections (\thesection) to be started by the serial of the part (\thepart) in which a section belongs, followed by a period (.) and then the serial number of the section in an Arabic numeral (\arabic{}). Without \renewcommand{}{}, sections under different parts will be numbered in the same way, like 1 or 2, which would be confusing to understand when a section is referred somewhere. An example of an article, where its sections are divided into parts and numbered part-wise, is shown in Table 19.11.

Table 19.11 Article dividing sections into parts and numbering them part-wise

<table>
<tr><td>LaTeX input</td><td>

```
\documentclass{article}
\makeatletter
\@addtoreset{section}{part}
\makeatother
\renewcommand{\thesection}{\thepart.\arabic{section}}
%
\begin{document}
\part{}\label{part:country}
\section{India}\label{sec:ind}
\subsection{Population of India}\label{sec:indpop}
\subsubsection{Per Capita Income in India}
%
\part{}\label{part:state}
\section{Delhi}\label{sec:del}
\subsection{Population of Delhi}\label{sec:delpop}
%
India is described in \S\ref{sec:ind} of Part~\ref{part:country} ...
Population of Delhi can be found in \S\ref{sec:delpop}.
\end{document}
```

</td></tr>
<tr><td>Output</td><td>

Part I

I.1 India

I.1.1 Population of India

I.1.1.1 Per Capita Income in India

Part II

II.1 Delhi

II.1.1 Population of Delhi

India is described in §I.1 of Part I … Population of Delhi can be found in §II.1.1.

</td></tr>
</table>

Hour 20

Book and Report

Preparation of a book or a report is similar to that of an article discussed in Hour 19. The only difference is that a book or a report contains a number of chapters, where each chapter is like an independent article. A book is prepared through the document-class **book** and a report through the document-class **report**. The structure of a report is similar to that of a book, expect some differences in the output. Therefore, the report preparation is not discussed here separately. The differences can be observed just by changing the document-class of a document from **book** to **report**, or from **report** to **book**. Note that an academic thesis can be prepared in any of the document-classes of **book** and **report**.

20.1 Template of a Book

The general template of a book is shown in Table 20.1 on the following page, where the book is produced in three parts – **\frontmatter** (front matter), **\mainmatter** (main matter), and **\backmatter** (back matter). The **\frontmatter** command covers the title, preface, contents, etc., with page numbering in Roman numerals and no numbering to the **\chapter{}** command. The main part of the book comes under the **\mainmatter** command with page and chapter numbering in Arabic numerals. Finally, appendix, bibliography, index, etc., are inserted under the **\backmatter** command without chapter numbering.

The default template given in Table 20.1 can be altered by user-defined templates, an example of which is shown in Table 20.2 on page 193 (if compiled, its output would be similar to this book to a large extent). The **twoside** option is used in **\documentclass[]{}** for producing the book on both sides of a page (which is the usual requirement in a book). The *cover page* is started with **\thispagestyle{empty}** for not assigning any page number to this page. The **\cleardoublepage** command, used after the *cover page* (also after the *Preface* and the *Table of Contents*), leaves the last even-numbered page blank, if the previous unit ends on an odd-numbered page, and starts the following unit on the next odd-numbered page (**\cleardoublepage** is not required

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_20

Table 20.1 General template of a book in the document-class **book**

```
\documentclass[11pt,a4paper]{book}
⋮
\title{\LaTeX\ in 24 Hours\\A Practical Guide for Scientific Writing}
\author{Dilip Datta}
%
\begin{document}
\frontmatter              % First part of the book
\maketitle
\chapter{Preface}
⋮
\tableofcontents
%
\mainmatter               % Second part of the book
\chapter{Introduction}
⋮
\chapter{Equation1}
⋮
\backmatter               % Last part of the book
\appendix
\chapter{Appendix}
⋮
\bibliographystyle{plain}
\bibliography{mybib}
\printindex
\end{document}
```

between two chapters as, by default, a chapter is started on an odd-numbered page). Before starting the *Preface*, **\pagenumbering{roman}** is used for numbering pages now onwards in Roman numerals. The *Preface* is inserted as a chapter with the **\chapter*{}** command (instead of **\chapter{}**) for not assigning a serial number to it. Next, the **\tableofcontents** command is used for automatically generating the contents list of the book. Finally, the actual chapters of the book are started, preceded by **\pagenumbering{arabic}** for numbering pages now onwards in Arabic numerals.

20.2 Book Preparation Using a Root File

Table 20.2 shows a template for preparing a book in a single input file. Since a book usually contains a large number of pages, practically it will be difficult to work with such a single file because of its huge size in accommodating the contents of a book. Hence, as a convenient way, a book is generally prepared in a number of input files of smaller sizes. Usually each part of a book, such as the title, preface, dedication,

Table 20.2 User-defined template of a book in the document-class **book**

```
\documentclass[a4paper,11pt,twoside]{book}
⋮
\begin{document}
% Cover page
\thispagestyle{empty}
\begin{center}
   {\Huge\bf \LaTeX\ in 24 Hours}\\[5mm]
   {\Large\bf A Practical Guide for Scientific Writing}
\end{center}
\cleardoublepage
% Preface
\pagenumbering{roman}
\chapter*{Preface}
The necessity for writing this book was felt long back ...
\cleardoublepage
% Contents
\tableofcontents
\cleardoublepage
% Starting chapters
\pagenumbering{arabic}
\chapter{Introduction}
Donald E. Knuth developed \TeX\ in the year 1977 ...
%
\chapter{Fonts Selection}
There are three modes for processing texts in \LaTeX\ ...
⋮
\appendix
\chapter{Appendix}
⋮
\bibliographystyle{plain}
\bibliography{mybib}
\printindex
\end{document}
```

chapters, appendix, and bibliography, is prepared in a separate input file and then all the individual files are compiled together to produce a single output file as the final book. Moreover, the preamble (i.e., the required packages, user-defined commands and environments, page formatting, or any other global setting) may also be prepared in a separate input file. The input file containing the list of bibliographic references is named with `bib` extension, say `mybib.bib`. All other input files are named with `tex` extension, such as, `preamble.tex`, `title.tex`, `preface.tex`, `chapter1.tex`, `chapter2.tex`, etc. Finally, all the individual input files are linked to a single root file, say `mybook.tex`, compilation of which produces the final book. An input file can be linked to the root file either through the **\input{}** or **\include{}** command (§20.4.2 on page 201 discusses the differences between the commands) with the name of the input file (without the extension) as the argument of the command. If prepared in the form shown in Table 14.2 on page 139, the bibliography file is included in the **thebibliography** environment using the **\input{}** command, otherwise (if prepared in the form shown in Table 15.2 on page 145) it is included through the **\bibliography{}** command following the **\bibliographystyle{}** command as shown in Tables 20.1 and 20.2. A sample preamble file, a few other files, and the root file of a book, similar to those of this book, are shown in Tables 20.3, 20.4, and 20.5.

Table 20.3 shows the preamble file (`preamble.tex`), most of the LaTeX instructions of which are already discussed in previous Hours, while others are self-explanatory. As mentioned above, by default a chapter is started on an odd-numbered page by leaving the previous even-numbered page blank, if required. However, it is

Table 20.3 Preamble of a book

```
% File name: preamble.tex
\documentclass[a4paper,11pt,twoside,openany]{book}
% Basic packages
\usepackage{float}                          % For new floating environments
\usepackage{stmaryrd,amssymb,amsmath}       % For mathematical symbols and equations
\usepackage{array}                          % For arrays of equations
\usepackage{epsfig,graphicx,subfigure}      % For inserting figures
\usepackage{wrapfig}                        % For wraping texts around tables and figures
\usepackage{tabularx}                       % For auto-adjusted column widths in tables
\usepackage{multirow}                       % For merging cells in tables
\usepackage{longtable}                      % For multi-page tables
\usepackage{rotating}                       % For rotating a page (landscape) or inclined texts
\usepackage{caption}                        % For adjusting captions of tables and figures
\usepackage{color}                          % For writing colored texts
\usepackage{setspace}                       % For adjusting line spacing
\usepackage{boxedminipage,fancybox}         % For boxed texts
\usepackage{shadow}                         % For creating shaded box
\usepackage{natbib}                         % For bibliographic references
\usepackage{varioref}                       % For referring through \vref{} & \vpageref{}
\usepackage{url}                            % For citing URL
\usepackage{makeidx}                        % For generating index
%
\makeindex                                  % Generate index
% Blank space adjustment
\abovecaptionskip                           % Skips extra space above a caption
\belowcaptionskip                           % Skips extra space below a caption
\raggedbottom                               % Top aligning a page leaving space at the bottom
% User-defined new commands
\definecolor{ugray}{gray}{0.25}             % User-defined gray color 'ugray'
\newcommand{\tgray}{\textcolor{ugray}}      % '\tgray{}' for writing in user-defined 'ugray'
\newcommand{\tred}{\textcolor{red}}         % '\tred{}' for writing in red
\newcommand{\vctr}[1]%
        {\mbox{\boldmath{$#1$}}}            % Prints x as a vector through \vctr{x}
\newtheorem{thm}{Theorem}                   % Environment 'thm' for writing theorems
\newtheorem{dfn}{Definition}                % Environment 'dfn' for writing definitions
\newtheorem{lem}{Lemma}                     % Environment 'lem' for writing lemmas
```

generally not desirable to leave any page blank in a book. The **openany** option to the **\documentclass[]{}** command works for that, which instructs to start a new chapter on the immediate next blank page, no matter whether odd- or even-numbered.

The sample files, containing the title, preface, Chapter 1, and appendix of a book, are shown in short form in Table 20.4. Note that neither the **\documentclass[]{}** command nor the **document** environment is required in these files. The **\documentclass[]{}** command is included in the preamble file, and the **document** environment is created in the root file in which these individual files are to be linked. The title page (`title.tex`) is written in the **titlepage** environment, which produces a page without any page number and heading. The **\vspace*{\fill}** command is inserted at the top and bottom of the file for vertically center aligning its contents. The Preface (`preface.tex`) is prepared as a chapter under the **\chapter*{}** command (**\chapter*{}** used instead of **\chapter{}** for preventing it from assigning any chapter number). The introduction (`chap_intro.tex`) is prepared as a normal chapter under the **\chapter{}** command (other chapters of a book are to be prepared in the same way). Even the appendix (`app_symb.tex`) is also prepared as a normal chapter like `chap_intro.tex`.

Table 20.4 Some sample input files of a book

```
% File name: title.tex
\vspace*{\fill}
\begin{titlepage}
\begin{center}
    {\Huge\bf \LaTeX\ in 24 Hours}\\[5mm]
    {\Large\bf A Practical Guide for Scientific Writing}
\end{center}
:
:
\end{titlepage}
\vspace*{\fill}
```

```
% File name: preface.tex
\chapter*{Preface}
The necessity for writing this book was felt long back, during my Ph.D
work, when I saw students and researchers struggling with \LaTeX\ for
writing their articles and theses ...
```

```
% File name: chap_intro.tex
\chapter{Introduction}\label{chap:intro}
%
\section{What is \LaTeX?}\label{sec:latex}
\LaTeX\ is a macro-package used as a language-based approach for
typesetting documents. Various \LaTeX\ instructions are interspersed with
the input file of a document, say myfile.tex, for obtaining the desired
output as myfile.dvi or directly as myfile.pdf ...
```

```
% File name: appsymb.tex
\chapter{Symbols and Notations}\label{app:symbol}
%
There are unlimited number of symbols and notations which may be required
to be used in different documents. Moreover, there exist many special
letters used in different languages. All such symbols and letters are to
be produced in a \LaTeX\ file through commands ...
```

Finally the root file (`mybook.tex`), where all the individual input files are linked, is shown in Table 20.5. Different segments of `mybook.tex` are as follows:

- ▷ At the very beginning, the preamble file (`preamble.tex`) is included through the **\input{**`preamble`**}** command, and then all other '`.tex`' files (without the `tex` extension) are included in the **document** environment through the **\include{}** command.
- ▷ The BIBTEX formatted bibliography file (`mybib.bib`) is inserted through the **\bibliography**{`mybib`} command, preceded by a bibliography style through the

Table 20.5 Root file linking individual input files of a book

```
% File name: mybook.tex
\input{preamble}
\begin{document}
    \begin{spacing}{1.2}
        % Cover Page, Title, Preface and Dedication
        \thispagestyle{empty}  \include{coverpage}  \cleardoublepage
        \pagenumbering{roman}
            \phantomsection\addcontentsline{toc}{chapter}{Title}
                \thispagestyle{empty}  \include{title}  \cleardoublepage
            \phantomsection\addcontentsline{toc}{chapter}{Dedication}
                \thispagestyle{empty}  \include{dedication}  \cleardoublepage
            \phantomsection\addcontentsline{toc}{chapter}{Preface}
                \thispagestyle{empty}  \include{preface}  \cleardoublepage
            % Contents, List of Tables and List of Figures
            \phantomsection\addcontentsline{toc}{chapter}{Contents}
                \thispagestyle{empty}  \tableofcontents  \cleardoublepage
            \phantomsection\addcontentsline{toc}{chapter}{List of Tables}
                \thispagestyle{empty}  \listoftables  \cleardoublepage
            \phantomsection\addcontentsline{toc}{chapter}{List of Figures}
                \thispagestyle{empty}  \listoffigures  \cleardoublepage
        % Chapters
        \pagenumbering{arabic}
            \include{chap_intro}
            \include{chap_font}
            \include{chap_format}
            \include{chap_table}
            .
            .
            .
    \end{spacing}
    % Appendix, Bibliography and Index
    \begin{spacing}{1.0}
        \begin{appendix}  \include{app_symb}  \end{appendix}

        \phantomsection\addcontentsline{toc}{chapter}{Bibliography}
            \bibliographystyle{plain}  \bibliography{mybib}  \clearpage

        \phantomsection\addcontentsline{toc}{chapter}{Index}
            \printindex  \cleardoublepage
    \end{spacing}
\end{document}
```

\bibliographystyle{} command, say \bibliographystyle{plain} for printing the bibliographic references in the plain style (refer §15.1 on page 141 for detail).

▷ By default the page number is not printed only on the first page of a chapter. Since the files of the cover page, title, and dedication are not prepared as chapters, the inclusion of each of these input files is preceded by the \thispagestyle{empty} command for not assigning any page number to these pages.

▷ The \cleardoublepage command is used in many places for starting the following unit on the next odd-numbered page.

▷ Since the title, dedication, and preface are not prepared as numbered units (the preface is prepared under \chapter*{} for preventing it from numbering), the inclusions of their input files are preceded by the \addcontentsline{}{}{} command for including them in the `Contents` list (refer §16.1.1 on page 153 for the use of \phantomsection before \addcontentsline{}{}{}). The arguments of \addcontentsline{}{}{} are, respectively, the location where the unit is to be entered (toc for `Contents` list), the type how the unit is to be treated (chapter) and the name of the unit to be printed in the `Contents` list (refer §16.1.1 on page 153 for detail).

▷ The \tableofcontents, \listoftables, and \listoffigures commands are used for automatically generating three lists – `Contents` list of sectional units, `List of Tables`, and `List of Figures`, respectively. Since no serial number is assigned, these lists are also included in the `Contents` list through the \addcontentsline{}{}{} command as mentioned above.

▷ An appendix is prepared as a normal chapter. It is identified as an appendix through the \appendix command or inserting the appendix input file in the appendix environment, which generates the label-word '`Appendix`', followed by the serial number of an appendix in an uppercase alphabet, like `A`, `B`, etc. (if the label-word '`Appendix`' is not generated but '`Chapter`', it can be obtained by inserting the \renewcommand{\chaptername}{Appendix} command before the inclusion of the appendix files). The \appendix command and appendix environment also cause sectional units of an appendix to be numbered properly, like `A.1`, `A.1.1`, etc. In order to assign the continued serial numbers to different appendices, the \appendix command is to be used only once before the inclusion of the first appendix file, or all the appendix files are to be inserted in a single appendix environment.

▷ The location for producing the index is specified through the \printindex command. The index is produced under the heading '`Index`'. Since the index is also not assigned any serial number, it is included in the `Contents` list through \addcontentsline{}{}{}.

▷ Two different line spacings are used in `mybook.tex`, through the spacing environment defined in the setspace package. From the cover page onwards, the line spacing is set at 1.2 up to the last chapter of the book, while the appendix and bibliography are produced in single line spacing.

▷ The \pagenumbering{roman} command is used before the inclusion of the title page (`title.tex`) for numbering the following pages in Roman numerals. Then the page numbering from the starting of the first chapter (`chap_intro.tex`) is changed to Arabic numerals through the \pagenumbering{arabic} command, which is continued up to the last page of the book.

20.3 Dividing a Book into Parts*

Like in articles where different sections can be divided into a number of parts as discussed in §19.2.6 on page 189, different chapters of a book can also be divided into a number of parts through the \part{} command. Each \part{} command generates the label-word '`Part`' followed by the serial number of the part in a uppercase Roman numeral, such as `Part I` or `Part II`. Any text in the argument of the \part{} command is printed as the heading of the part. Similar to articles, although divided into a number of parts, by default the chapters of the book are assigned continuous serial numbers irrespective of the parts in which they belong. For numbering the chapters part-wise, the following four lines of commands may be inserted in the preamble:

```
\makeatletter
\@addtoreset{chapter}{part}
\makeatother
\renewcommand{\thechapter}{\thepart.\arabic{chapter}}
```

where \@addtoreset{}{} command resets the chapters to be numbered part-wise. The \renewcommand{}{} command here redefines the numbering of chapters (\thechapter) to be started by the serial of the part (\thepart) in which a chapter belongs, followed by a period (.) and then the serial number of the chapter in an Arabic numeral (\arabic{}). Without this \renewcommand{}{} command, the chapters under different parts will be numbered in the same way, like `Chapter 1` or `Chapter 2`, which would be confusing to understand if a chapter is referred somewhere.

20.4 Compilation of a Book

Since a book is generally composed of a number of chapters, one or more appendices, bibliography, and index, its compilation would be a combination of those discussed in §15.4 on page 149 and §16.2.3 on page 160. As an example, the command-prompt compilation of the book '`mybook.tex`' of Table 20.5 would involve the following five lines of commands (refer §20.4.1 on the next page for detail):

```
$ latex mybook
$ bibtex mybook
$ makeindex mybook
$ latex mybook
$ latex mybook
```

The command ‘latex mybook’ in the first line compiles mybook.tex as well as all other .tex files included in mybook.tex. It also generates the output in the form of mybook.dvi. However, the bibliography and index remain uncompiled. These are required to be compiled separately, which are done through the commands bibtex and makeindex in the second and third lines, respectively. Then the command of the first line is repeated in the fourth line for linking all the compilation. As the last step, the same command is repeated once again in the fifth line for producing the complete and final output file ‘mybook.dvi’.

20.4.1 Executable File for Compiling a Book*

It is seen in §1.4 on page 4 that only one command is required to compile a simple document in a command-prompt. However, the number of commands increases with the increasing number of provisions. Section 15.4 on page 149 and §16.2.3 on page 160 show that four lines of commands are required if any of bibliography and index is included in a document. The number of commands is increased to five if both bibliography and index are included. This is shown on this page above at the time of compiling a book. Since a document may need to be compiled a number of times during its preparation, it may become a cumbersome job to type all these commands repeatedly. Therefore, all the commands may be stored in a separate compilation file, which can be run to compile all the commands stored in it. The compilation file, named ‘compile’ without any extension, prepared for compiling the book having the root file named as mybook.tex, is given in Table 20.6. The line numbers are

Table 20.6 Compilation file for compiling a book in command-prompt

```
1   rm *.aux *.log *.dvi *.blg *.bbl *.idx *.ilg *.ind *.toc *.lot *.lof

2   latex mybook
3   bibtex mybook
4   makeindex mybook
5   latex mybook
6   latex mybook

7   dvipdf mybook.dvi

8   rm *.aux *.log *.dvi *.blg *.bbl *.idx *.ilg *.ind *.toc *.lot *.lof
```

included for the purpose of explanation only. The commands in lines 2–6 are used to compile the book. The command of line 2 compiles mybook.tex as well as all other

.tex files included in mybook.tex, while the commands of lines 3 and 4 compile the bibliography (if prepared through the BIBTEX program) and index, respectively. The commands of lines 5 and 6 (same with that of line 2) link all the three compilations (latex, bibtex, and makeindex) and produce the final output. Finally, the command of line 7 generates the mybook.pdf file. During the entire compilation process, a number of intermediate files are generated, samples of which are shown in Table 20.7.

Table 20.7 Intermediate files generated during the compilation of a book

Command	Location	Generated files
latex mybook	Line 2 in Table 20.6	mybook.aux, coverpage.aux, title.aux, preface.aux, dedication.aux, chap_intro.aux, chap_font.aux, chap_format.aux, ..., mybook.log, mybook.dvi
bibtex mybook	Line 3 in Table 20.6	mybook.bbl, mybook.blg
makeindex mybook	Line 4 in Table 20.6	mybook.idx, mybook.ilg, mybook.ind
\tableofcontents	mybook.tex	mybook.toc
\listoftables	mybook.tex	mybook.lot
\listoffigures	mybook.tex	mybook.lof

Information about the references required in a .tex file is written in an auxiliary file having the same name but with aux extension. The transcript file with log extension (having the same name with the root file of the book) records all the execution information, such as the names of the files read, the numbers of the pages processed, warning and error messages, and other pertinent data. Most of the information of the .log file is displayed on the screen also. The BIBTEX program generates the .bbl and .blg files, where sorted bibliographic references are stored. The files with idx, ind, and ilg extensions are index-related files, where indexed items and associated information are written. Other LATEX output files are related with the contents list, List of Tables, and List of Figures, having extensions toc, lot, and lof, respectively. The final output is written in a *device independent* file, having dvi extension, with a resolution better than a thousandth of an inch.

Coming back to Table 20.6, mybook.dpf file is produced in line 7 from the final mybook.dvi file. Then all the intermediate files, mentioned in Table 20.7, are removed in line 8 using the linux command 'rm'. Note that, if the execution of the compile file is terminated in between due to some bug in the LATEX input files, sometimes the compiler may fail to overwrite many intermediate files listed in Table 20.7, as a result of which the same bug would be shown even after correcting it. This is the reason why the intermediate files, if any already exists, are removed in line 1 also, prior to the compilation of the book using the commands of lines 2–6.

The compilation file 'compile' shown in Table 20.6 can be made executable using the 'chmod 777 compile' command (only once in a computer) in a command-prompt terminal, and then it can be run using the ./compile command.

20.4.2 Partial Compilation of a Book*

It is stated in §20.2 on page 192 that a .tex file can be included in the root file of a book using either the \include{} or \input{} command interchangeably. The \include{} command has an advantage over the \input{} command. Using the \includeonly{} command in the preamble, LaTeX can be instructed to compile only some selective files out of those included in the root file through the \include{} command, e.g., \includeonly{intro, font, format} can be used for compiling intro.tex, font.tex, and format.tex only. The \includeonly{} command is useful at the time of editing a large-size document like a book, where changes are to be checked in modified file(s) only. Note that if the number of pages of the document gets changed upon editing, the use of \includeonly{} may result in mismatched page numbers, thus restricting the use of the command in the final version of the document.

Hour 21

Slide Preparation I

The LaTeX platform is applicable for preparing slides also, which can be presented like those prepared in popularly known Microsoft PowerPoint package. In LaTeX, slides can be prepared through the document classes of **seminar**, **slides**, **prosper**, or **beamer**. All LaTeX macros (packages, commands, and environments), used in other document classes as discussed in previous Hours, are applicable in these document classes also. However, because of widespread application, only the document-class **beamer** is discussed here.

The BEAMER package, which defines the **beamer** document-class, is generally included in most of the standard LaTeX distributions mentioned in §1.3 on page 2. Like normal LaTeX input files, an input file with the document-class **beamer** also has the `tex` extension and it can be prepared in any LaTeX editor. Further, a **beamer** document-class file can be compiled using the same set of commands discussed in §1.4, §14.3 or §15.4, as applicable.

21.1 Frames in Presentation

In the document-class **beamer**, a presentation consists of a number of frames (or slides). A frame is created either by the **\frame[][]{}** command or the **frame** environment as **\begin{frame}[][]**...**\end{frame}**. The contents of a frame is inserted in the mandatory argument of the **\frame[][]{}** command or in the body of the **frame** environment. On the other hand, the first optional argument of a frame is to specify the piece-wise presentation of items as discussed in §22.1 on page 217, while the second optional argument is for other options for a frame (two options separating by a comma). A frame generally consists of some or all of the following eight components:

(1) **Headline and footline**: These are similar to header and footer of standard LaTeX, but generated automatically by the chosen theme (§21.4 on page 209 discusses themes) for displaying presentation-related various information.

(2) **Sidebars**: Sidebars are generated automatically by the chosen theme on either side for displaying mainly the table of contents of the presentation.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_21

(3) Navigation bars: Navigation bars are also produced automatically by the chosen theme mainly for the following two purposes:

(a) At any point of time during the presentation, the audience can see how much of the talk have been covered and what is yet to come.
(b) If required, the presenter can jump to a particular frame by clicking on the corresponding link.

(4) Navigation symbols: Eight number of default navigation symbols are shown by small icons in light gray color in the bottom right corner of every slide. These from left to right are known as the slide icon, frame icon, subsection icon, section icon, presentation icon, appendix icon, back and forward icons, and search icon. Each of the slide, frame, subsection, and section icons is preceded by a left arrow and followed by a right arrow. A click on the left arrow will lead, respectively, to the previous slide, the last slide of the previous frame, the last slide of the previous subsection, or the last slide of the previous section. Similarly, a click on the right arrow will lead, respectively, to the next slide, the first slide of the next frame, the first slide of the next subsection, or the first slide of the next section.

(5) Logo: A logo can be printed globally in all the frames through \logo{} in the preamble. It can contain a piece of plain texts or a figure insertion command, e.g., \logo{\includegraphics[width=8mm]{tul}} (§21.5.1 on page 213 discusses in detail).

(6) Frame title: A title and a subtitle can be assigned to a frame using the \frametitle{} and \framesubtitle{} commands in the mandatory argument of \frame[][]{} or inside the frame environment.

(7) Background: Each frame has a background, which consists of a background canvas and the main background. The background canvas is a big rectangle filling the whole frame, on which the main background and other things appear.

(8) Frame contents: The contents of a frame could be LaTeX supported any text, except the \verb" " command or verbatim environment[1], but including frame title and subtitle stated above. The contents of a frame are inserted in the mandatory argument of \frame[][]{} or inside the frame environment. By default the contents of a frame (except the title and subtitle) are vertically center aligned. This default alignment can be changed by the options t for top alignment, c (default) for vertically center alignment and b for bottom alignment. A vertical alignment option may be assigned to \documentclass[]{beamer} for global effect in all frames. Alternatively, it can be used as an option to a particular frame for local effect only, e.g., \frame[t]{} or \begin{frame}[t].

[1] Verbatim texts can be inserted in a frame through the \verb" " command or verbatim environment using the containsverbatim option (e.g., as \frame[containsverbatim]{}) defined in the fancyvrb package (however, it does not work under any overlay specification as discussed in Hour 22).

21.2 Sectional Units in Presentation

The frames of a presentation, as stated in §21.1, may be put under various sections and subsections produced through **\section[]{}** and **\subsection[]{}**. Unlike in standard LaTeX, **\section[]{}** and **\subsection[]{}** here do not create any heading at their positions, rather they add entries in the table of contents and navigation bars. Based upon the chosen theme (§21.4 on page 209 discusses themes), generally the full headings under the mandatory arguments of **\section[]{}** and **\subsection[]{}** are added in the table of contents, while the short headings under their optional arguments are added in navigation bars. In some cases, however, the subsections may be reflected by some symbols, such as a small circle.

21.3 Presentation Structure

Most of the presentations mainly contain the following types of pages/frames, which can be put under different sections and subsections for making their entry in the table of contents and navigation bars:

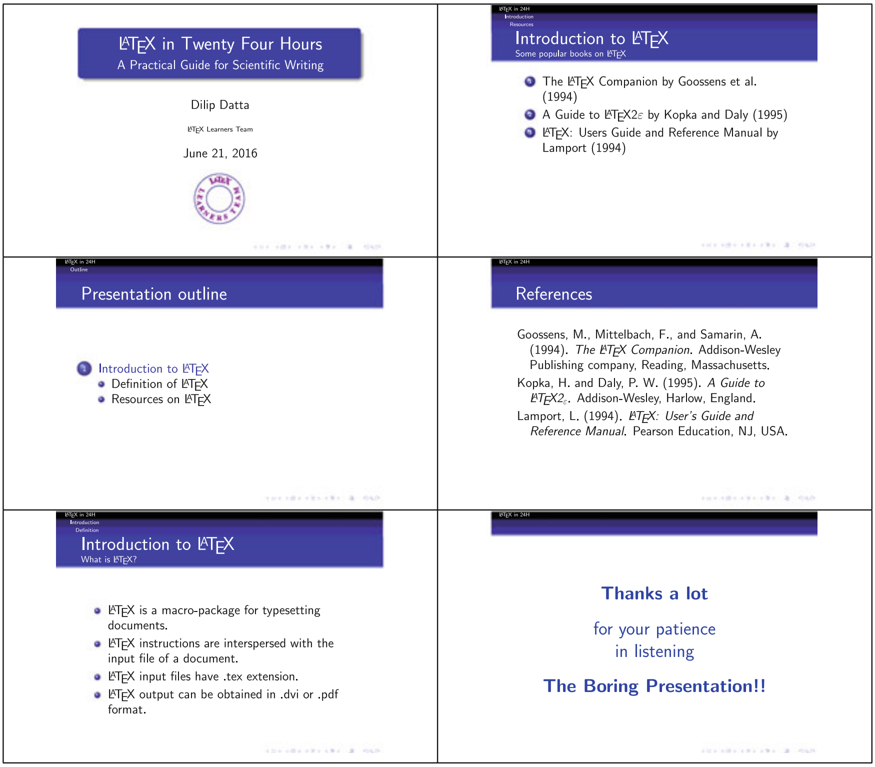
▷ Title page
▷ Table of contents
▷ Presentation materials
▷ Appendix
▷ Bibliography
▷ Thanks giving

Such a simple presentation input file consisting of six frames under the **JuanLesPins** presentation theme (§21.4 on page 209 discusses themes) is shown in Table 21.1 on the next page and its output column-wise in Table 21.2 on page 207. Just to show their applications, frames are created through both the **\frame[]{}** command and **frame** environment. The input file is started by **\documentclass{}** with the **beamer** document-class, followed by the **JuanLesPins** presentation theme loaded through **\usetheme{}**. Next, the **natbib** package is loaded through **\usepackage[]{}** with **sort** option for generating a sorted bibliographic list (Table 21.3 on page 209 discusses in detail). The other major component in the preamble is the title page related commands (refer §21.3.1 on page 207 for detail). Then the frames, containing different components of a presentation under various sectional units, are prepared in the **document** environment. The title page is generated in the very first frame. The next frame, producing the table of contents through **\tableofcontents**, is put under **\section*{}** for excluding its entry in the table of contents (§22.1.1 on page 217 discusses various options in the table of contents). Then the main contents of a presentation are inserted in intermediate frames, as shown in Table 21.1 producing frames 3 and 4 under one section and two subsections. Finally, the bibliographic reference list and thanks giving may be inserted in the ending frames. These two frames are

Table 21.1 A simple presentation input file

```
\documentclass{beamer}
\usetheme{JuanLesPins}
\usepackage[sort]{natbib}
% Components of the title page
\title[\LaTeX\ in 24H]{\LaTeX\ in Twenty Four Hours}
\subtitle{A Practical Guide for Scientific Writing}
\author[D. Datta]{Dilip Datta}
\institute[\LaTeX-LT]{\LaTeX\ Learners Team}
\date[L24H :: 21-06-2016]{June 21, 2016}
\titlegraphic{\includegraphics[width=20mm]{logo_LA}}
%
\begin{document}
% Frame 1
\frame[plain]{\titlepage}
% Frame 2
\section*{Outline}
\frame[t]{ \frametitle{Presentation outline} \tableofcontents }
% Frames 3 and 4
\section[Introduction]{Introduction to \LaTeX}
\subsection[Definition]{Definition of \LaTeX}
\frame[t]
{   \frametitle{Introduction to \LaTeX} \framesubtitle{What is \LaTeX?}
    \begin{itemize}
    \item \LaTeX\ is a macro-package for typesetting documents.
    \item \LaTeX\ instructions are interspersed with ...
    \item \LaTeX\ input files have .tex extension.
    \item \LaTeX\ output can be obtained in .dvi or .pdf format.
    \end{itemize}
}
\subsection[Resources]{Resources on \LaTeX}
\begin{frame}[t]
  \frametitle{Introduction to \LaTeX} \framesubtitle{Some popular books on \LaTeX}
  \begin{enumerate}
  \item The \LaTeX\ Companion by \citet{Goossens-etal-1994}
  \item A Guide to \LaTeX2$_\varepsilon$ by \citet{Kopka-Daly1997}
  \item \LaTeX: User's Guide and Reference Manual by \citet{Lamport-1994}
  \end{enumerate}
\end{frame}
% Frame 5
\section*{}
\begin{frame}[t]
  \frametitle{References}
  \bibliographystyle{apalike} \bibliography{lswbib}
\end{frame}
% Frame 6
\section*{}
\begin{frame}
  \begin{center}
  \Large{\bf\textcolor{blue}{Thanks a lot}}\\[5mm] ... \end{center}
\end{frame}
\end{document}
```

Table 21.2 Slides under the **JuanLesPins** presentation theme for input file of Table 21.1



prepared under **\section*{}** without any argument so as to skip their entry in the table of contents as well as in headline/footline and sidebars (without **\section*{}**, these frames may be shown wrongly under the previous sectional unit).

Note that the input file shown in Table 21.1 will be used, inserting additional sections and frames with required changes, in all examples of this Hour as well as those in Hour 22.

21.3.1 Title Page

The title page of a presentation is produced through the **\titlepage** command. In order to produce the title page in a frame, **\titlepage** is to be put in the **frame** environment, or in **\frame[][]{}** as shown in Table 21.1 as Frame 1. The **plain** option is used here for omitting headline/footline and sidebars in the frame (§21.5.5 on page 215 discusses in detail).

The title page generally contains sequentially a title, a subtitle, list of authors (or presenters), affiliations of the authors, presentation date, and a symbolic affiliation, which are generated in the preamble through the **\title[]{}**, **\subtitle{}**, **\author[]{}**, **\institute[]{}**, **\date[]{}**, and **\titlegraphic{}** commands, respectively.

1. The title of the presentation is generated as **\title[**stitle**]{**ftitle**}**, where the optional stitle is a short title to be used in the headline/footline (if generated) and the mandatory ftitle is the full title to be produced in the title page. The **** command is allowed for splitting the title in multiple lines.
2. The **\subtitle{}** command may be used for producing a second title, in a smaller size, below the main title.
3. The list of authors is produced as **\author[**sname**]{**fname**}**, where the optional sname is a short name to be used in the headline/footline (if generated) and the mandatory fname is the full list of authors to be produced in the title page. In **\author[]{}**, two authors are separated by **\and**. If authors have different affiliations, their names are to be suffixed by **\inst{}** with affiliation number in its argument, e.g., **\author[**Datta et al.**]{**D. Datta**\inst{**1**}** **\and** P.K. Nath**\inst{**2**}** **\and** S. Dutta**\inst{**2**}}**.
4. The affiliation of authors is generated as **\institute[**saff**]{**faff**}**, where the optional saff is a short affiliation for headline/footline (if produced) and the mandatory faff is the full affiliation for the title page. Each of multiple affiliations marked in **\author[]{}** is to be prefixed by the corresponding **\inst{}**, and two affiliations are to be separated by **\and**. If required, line break may also be obtained using ****. For example, **\institute[**TU **\&** NITS**]{\inst{**1**}**Tezpur University, Tezpur**** **\and** **\inst{**2**}**National Institute of Technology, Silchar**}**.
5. The presentation date may be displayed through **\date[**sdate**]{**fdate**}**, where the optional sdate is a short date for headline/footline (if produced) and the mandatory fdate is the full date for the title page, e.g., **\date[**20/06/16**]{**June 20, 2016**}**.
6. Finally, an affiliation logo may be inserted as **\titlegraphic{**tgraph**}**, where tgraph could be a piece of plain texts or a figure insertion command as shown in Table 21.1.

21.3.2 Presentation Contents

The actual contents of a presentation generally come in intermediate frames between the frames of the title page and bibliographic reference page as discussed in §21.3.1 and §21.3.3, respectively. The presentation contents of a frame may be produced under a frame title and a frame subtitle, which are generated through the **\frametitle{}** and **\framesubtitle{}** commands. Such examples are shown in Frame 3 and Frame 4 of Table 21.1, where two frames are produced under the same frame title but different frame subtitles.

Contents of a presentation could be LaTeX supported any text except **\verb" "** command or **verbatim** environment. Since the contents of a presentation are generally presented point-wise instead of in paragraph form, the contents of a frame are usually arranged in a listing environment, such as the **enumerate**, **itemize**, and **description** environments (refer Frames 3 and 4 in Table 21.1, where the **itemize** and **enumerate** environments are used for the same).

21.3.3 Bibliographic Reference Page

The BEAMER class seeks a bibliographic reference list to be prepared in the **thebibliography** environment discussed in Hour 14. However, the BIBTEX program discussed in Hour 15 is also accepted for the same. In that case, any error or warning message, like ‘ `\newblock undefined`’, may simply be ignored.

Most of the BEAMER themes (refer §21.4 for detail) put the bibliographic reference list under the default heading `References`. If not, the same may be generated through **\frametitle{ }** as the title of the corresponding frame. Further, for a long list of bibliographic references, the **allowframebreaks** option may be used in the frame for splitting it over multiple slides (§21.5.6 on page 215 discusses in detail).

21.4 Appearance of a Presentation (BEAMER Themes)

It is always desirable to make a presentation attractive as much as possible. The appearance of a presentation in the **beamer** document-class can be controlled by five types of themes, which are presentation theme, color theme, font theme, inner theme, and outer theme. A presentation theme generally controls every single detail of the appearance of a presentation. Since every presentation theme uses a default set of other four themes, normally no other theme is required to be specified if a presentation theme is chosen. In order to alter the default setting of a presentation theme, still separate color, font, inner, or outer theme may be used as per requirement or choice.

The above five types of themes are to be loaded in the preamble, respectively, as **\usetheme[**`oname`**]{**`tname`**}**, **\usecolortheme[**`oname`**]{**`tname`**}**, **\usefonttheme[**`oname`**]{**`tname`**}**, **\useinnertheme[**`oname`**]{**`tname`**}**, and **\useoutertheme[**`oname`**]{**`tname`**}**, where mandatory `tname` is the name of the chosen presentation/color/font/inner/outer theme and optional `oname` is an option to `tname`. Combining all the five types into a single one, a theme can also be loaded as a package as **\usepackage{beamertheme**`tname`**}** by appending the fixed word **beamertheme** with `tname`, e.g., **\usepackage{beamerthemedefault}** for loading the **default** theme, or **\usepackage{beamerthemeBerkeley}** for loading the **Berkeley** theme.

21.4.1 Presentation Theme

Various types of available presentation themes are listed in Table 21.3 on the following page. Effects of some of these presentation themes can be noticed in different examples as follows: **JuanLesPins** in Table 21.2, **Frankfurt** in Table 22.3, **Hannover** in Table 22.7, **Berlin** in Table 22.8, **Warsaw** in Table 22.9, **Madrid** in Table 22.11, **Singapore** in Table 22.12, and **Boadilla** in Table 22.14.

Navigational bars appearing under different presentation themes may occupy a large amount of space of a frame, like in displaying sectional units in different lines. If space is crucial, the **compress** option may be used in **\documentclass[ ]{beamer}** for making navigation bars globally as small as possible, and also to compress sectional units in one line.

Table 21.3 Various types of presentation themes under the BEAMER package

Type	Theme	Function
Without navigation bars	**default**	It is a sober theme that uses minimal color or font variations.
	boxes	Height of headline/footline boxes can be controlled through the options **headheight** and **footheight**, e.g., **headheight=**10**pt**.
	Bergen	It is based on **inmargin** and **rectangles** inner themes (refer Table 21.6).
	Boadilla	Gives a lot of information in a little space. The **secheader** option may be used for showing the current section and subsection in headline.
	Madrid	Similar to the **Boadilla** theme, except the use of stronger colors. It also support the option of the **Boadilla** theme.
	AnnArbor	Similar to the **Boadilla** theme, but uses colors of the University of Michigan.
	CambridgeUS	Similar to the **Boadilla** theme, but uses colors of MIT.
	EastLansing	Similar to the **Boadilla** theme, but uses colors of Michigan State University.
	Pittsburgh	A sober theme with right-flushed frame titles.
	Rochester	A dominant theme. The height of the frame title bar can be controlled through the **height** option, e.g., **height=**10**mm**.
With a tree like navigation bar	**Antibes**	Navigation bars are shown at the top in separate rectangular elements.
	JuanLesPins	Similar to the **Antibes** theme, but has a much smoother appearance.
	Montpellier	A sober theme giving basic navigational hints.
With a table of contents sidebar	**Berkeley**	A sidebar shows the table of contents with the current entry highlighted. Available options to the theme include **hideallsubsections** for suppressing all subsections in the sidebar, **hideothersubsections** for suppressing all subsections other than those of the current section, **right** for putting the sidebar on the right side (the default is **left**), and **width** for setting the width of the sidebar, e.g., **width=**20**mm** (**width=**0**mm** eliminates the sidebar).
	PaloAlto	Similar to the **Berkeley** theme with the same options applicable here also.
	Goettingen	A full table of contents is shown in a sidebar, and the options of the **Berkeley** theme are applicable here also.
	Marburg	Similar to the **Goettingen** theme with the same options applicable here also.
	Hannover	A sidebar is shown on the left side, and the frame title is right-flushed. The options **hideallsubsections**, **hideothersubsections** and **width**, as mentioned in the case of the **Berkeley** theme, are applicable here also.
With a mini frame navigation	**Berlin**	The headline and footline show a lot of information. The **compress** option may also be used to display the information of the headline in a single line.
	Ilmenau	Similar to the **Berlin** theme with the same options applicable here also.
	Dresden	Similar to the **Berlin** theme with the same options applicable here also.
	Darmstadt	There is a strong separation between the navigational upper part and the informational main part.
	Frankfurt	A variation of the **Darmstadt** theme, which is slightly less cluttered by leaving out the subsectional information.
	Singapore	The navigation is not so dominating one.
	Szeged	A sober theme with a strong dominance by horizontal lines.
With section and subsection tables	**Copenhagen**	Shows compressed information about the current section and subsection at the top, and about the title and author at the bottom.
	Luebeck	A variation of the **Copenhagen** theme.
	Malmoe	A more sober variation of the **Copenhagen** theme.
	Warsaw	A dominant variation of the **Copenhagen** theme.

21.4.2 Color Theme*

The appearance of a presentation can be drastically changed using different color themes, which are listed in Table 21.4 on the next page. An outer color theme is used to change the colors of elements in outer themes (§21.4.5 on page 213 discusses outer

Table 21.4 Various color themes under the BEAMER package

Type	Theme	Function
Default and special-purpose color	**default**	Uses a little special colors and even less backgrounds.
	sidebartab	Changes colors in sidebars in a way that the current entry in the table of contents is highlighted by a different background.
	structure	Offers a convenient way of changing the color of the foreground structure. Various options available to the theme include **rgb={**`r`**,**`g`**,**`b`**}** with `r`, `g` and `b` as the decimal values of red, green and blue between 0 and 1, e.g., **rgb={**`0.5`**,**`0.2`**,**`1`**}**; **RGB={**`r`**,**`g`**,**`b`**}** same with **rgb**, except the numbers ranging between 0 and 255, e.g., **RGB={**`120`**,**`0`**,**`70`**}**; **cmyk={**`c`**,**`m`**,**`y`**,**`k`**}** with `c`, `m`, `y` and `k` as the values of cyan, magenta, yellow and black between 0 and 1, e.g., **cmyk={**`1`**,**`0`**,**`0.3`**,**`0.5`**}**; **cmy={**`c`**,**`m`**,**`y`**}** same with **cmyk** without any black component; **hsb={**`h`**,**`s`**,**`b`**}** with `h`, `s` and `b` as the values of hue, saturation and brightness between 0 and 1, e.g., **hsb={**`0.2`**,**`1`**,**`0.5`**}**; and **named={**`cname`**}** with `cname` as the name of a predefined color.
Complete color	**albatross**	Uses yellow on blue as the main colors, and also a slightly darker background for blocks (this background can be removed by additionally loading the **lily** color theme). The **overlystylish** option may be used to this theme for obtaining a background canvas.
	beetle	Uses white and black texts on gray background (white text is used for special emphasis and black text for normal cases). The outer stuff, like the headline/footline, however, uses a bluish color.
	crane	Uses the colors of Lufthansa, whose logo is a crane.
	dove	It is nearly black and white, which uses grayscale in certain unavoidable cases, but never a color. It also produces alerted texts in boldface fonts.
	fly	It is similar to the **beetle** theme, and uses white/black/gray throughout.
	monarca	It is based on the colors of the Monarch butterfly.
	seagull	Extensively uses different shades of gray color.
	wolverine	It is based on the colors of the University of Michigan's mascot, a wolverine.
	beaver	It is based on the colors of MIT's mascot, a beaver.
	spruce	It is based on the colors of the Michigan State University.
Inner color	**lily**	It is used mainly to restore the default colors by removing block colors setup by another theme, i.e., it removes all background colors of blocks.
	orchid	Uses white-on-dark block titles, red background for alerted blocks, and green background for example blocks.
	rose	Uses nearly transparent backgrounds for both block titles and block bodies.
Outer color	**whale**	Uses white-on-dark palettes for headline, footline, and sidebar.
	seahorse	Uses near-transparent backgrounds for headline, footline, and sidebar.
	dolphin	Its effect lies somewhere in between the **whale** and **seahorse** themes.

themes), such as headline, footline, and sidebar. On the other hand, an inner color theme specifies the colors of elements in inner themes (§21.4.4 on the next page discusses inner themes), specifically the colors used for blocks. If a color theme is used to change the default inner colors of a presentation theme or another color theme, it should be loaded after the other theme.

21.4.3 Font Theme*

The **beamer** document-class contains a set of font themes, which can be used to change certain font attributes. Such available themes are listed in Table 21.5.

Table 21.5 Various font themes under the BEAMER package

Theme	Function
default	Uses sans serif fonts for all texts of the presentation.
serif	Uses default serif fonts for all texts of the presentation. Some options may also be used, such as **stillsansserifmath** along with the **stillsansseriftext** option for producing mathematical texts in sans serif fonts; **stillsansserifsmall** for producing "small" texts in sans serif fonts, particularly the texts in headline, footline and sidebars; **stillsansseriflarge** for "large" texts in sans serif fonts, like the presentation or frame title; **stillsansseriftext** for normal texts in sans serif fonts; and **onlymath** for mathematical texts in serif fonts.
structurebold	Titles and texts in headline, footline and sidebars are produced in boldface fonts. The options that may be used in this theme are **onlysmall** for producing "small" texts in headline, footline and sidebars (but not titles) in boldface fonts; and **onlylarge** for producing "large" texts in boldface fonts, particularly in the main title, frame titles, and section entries in the table of contents.
structureitalicserif	Similar to the **structurebold** font theme, except texts are produced by serif fonts in boldface and italics modes. The options of the **structurebold** theme are supported by this theme also.
structuresmallcapsserif	Similar to the **structurebold** font theme, excepts texts are produced by serif fonts in small caps mode. The options of the **structurebold** theme are supported by this theme also.

21.4.4 Inner Theme*

An inner theme controls the appearance of the elements occurring inside the main texts of a frame, such as the title, listing/theorem/proof environments, figures, tables, foot notes, and bibliography entries. Various available inner themes are listed in Table 21.6.

Table 21.6 Various inner themes under the BEAMER package

Theme	Function
default	An item in the **itemize** environment start with a little triangle.
circles	An item in the **itemize** and **enumerate** environments, as well as an entry in the table of contents starts with a small circle.
rectangles	An item in the **itemize** and **enumerate** environments, as well as an entry in the table of contents starts with a small rectangle.
rounded	An item in the **itemize** and **enumerate** environments, as well as an entry in the table of contents starts with a small ball. The **shadow** option may be used to add a shadow to all the blocks.
inmargin	A block title or item marking is shown on the left side and its body on the right side.

*21.4.5 Outer Theme**

An outer theme controls the appearance of the elements occurring around the main texts of a frame, such as the headline, footline, sidebar, logo, and frame title. In other words, an outer theme controls roughly the overall layout of a frame. Various available outer themes are listed in Table 21.7.

Table 21.7 Various outer themes under the BEAMER package

Theme	Function
default	There is no headline/footline, and the frame title is left flushed.
infolines	Headline shows the current section and subsection, while footline shows the author, institution, presentation title, date, and frame count.
miniframes	Headline shows a navigational bar containing sections, and below every section a small clickable circle against each frame of the section or its subsections. Just below the navigation bar, the title of the current subsection is displayed, which can also be suppressed using the option **subsection=false**. Further, a footline can also be produced with the option **footline=**`fval`, where `fval` could be **authorinstitute** to show author and institute, **authortitle** to show author and title, **institutetitle** to show institute and title, and **authorinstitutetitle** to show author, institute and title.
smoothbars	Headline is similar with that under the **miniframes** theme. Showing the subsections in the headline can be suppressed with the option **subsection=false**. Further, footlines of the **miniframes** theme can be obtained by additionally loading this theme also.
sidebar	A sidebar is shown containing a small table of contents with the current section or subsection highlighted, and the frame title is vertically centered occupying the same amount of space in all frames. Options that can be used include **height=**`hdim` with `hdim` specifying the height of the space for the frame title (**height=**`0`**pt** will instruct to occupy only the required space), **hideothersubsections** to cause all subsections except those of the current section to be suppressed in the table of contents, **hideallsubsections** to suppress all subsections in the table of contents, **right** to put the sidebar on the right side (default is **left**), and **width=**`wdim` with `wdim` specifying the width of the sidebar (**width=**`0`**pt** suppresses it completely).
split	Sections are shown on the left side of the headline, while subsections of the current section on its right side. The footline shows the author on the left side and the presentation title on the right side.
shadow	Extends the **split** theme by putting a horizontal shading behind the frame title and adding a little shadow at the bottom of the headline.
tree	Headline shows a navigational tree containing the presentation title, current section, and current subsection in three different lines. The option **hooks** may be used to draw little hooks in front of section and subsection entries.
smoothtree	Similar to the **tree** theme, except background colors changing smoothly.

21.5 Frame Customization*

The frames of a presentation can be customized in different ways, such as the logo position, font type, global and local frame sizes, etc.

21.5.1 Logo in Frames

A logo is produced in the title page through **\titlegraphic{}** as stated in §21.3.1 on page 207, or globally in all the frames through **\logo{}** as stated in §21.1 on page 203. The position of the logo in the frames, inserted through **\logo{}**, is determined by the chosen theme discussed in §21.4 on page 209.

If the logo produced by **\logo{}** is unsatisfactory, the same can also be produced indirectly as the optional argument of one of the title page generating commands (refer §21.3.1 for detail), e.g., **\institute[**`TU`**\quad \epsfig{file=**`logo_tu.eps`**,width=**`10`**mm}]{}** for producing a graphical logo in the footline along with the institutional name. Note at this juncture that a command with an optional argument cannot be used in an optional argument of another command[2], e.g., in the above case the graphical logo cannot be produced in **[ ]** of **\institute[ ]{}** through **\includegraphics[ ]{}** command specifying the size of the figure in **[ ]** of **\includegraphics[ ]{}**.

21.5.2 Font Type

Even if a font theme is used as discussed in §21.4.3 on page 212, some font-related changes may still require some document-class specific options or special packages.

In regard to font size, its default value in the BEAMER class is 11 pt. If required, either a smaller font size may be opted in **\documentclass[ ]{beamer}** to accommodate more texts on each slide, or a larger font size to fill up the slides. Various font sizes defined in the BEAMER class are **8pt**, **9pt**, **10pt**, **smaller**, **11pt** (default), **12pt**, **bigger**, **14pt**, **17pt**, and **20pt** (some of these font sizes may require to load the **extsizes** package).

In regard to font family, by default the BEAMER class uses the Computer Modern fonts. This can be altered by loading appropriate package in the preamble, e.g., the **mathptmx** package for the Times font family, or the **helvet** package for the Helvetica font family.

21.5.3 Frame Size

The default size of a BEAMER frame is 128 mm×96 mm, whose aspect ratio is 4:3. This default size can be altered through the **aspectratio** option to **\documentclass[ ]{}** as **\documentclass[\aspectratio=**`arval`**]{beamer}**, where `arval` is the chosen value of the aspect ratio. The available values of `arval` are listed in Table 21.8 on the next page.

[2] A command with an optional argument cannot be used in an optional argument of another command.

Table 21.8 Available frame sizes in the BEAMER package

Value of **aspectratio**	Aspect ratio	Frame size
`1610`	16:10	160 mm × 100 mm
`169`	16:9	160 mm × 90 mm
`149`	14:9	140 mm × 90 mm
`141`	1.41:1	148.5 mm × 105 mm
`54`	5:4	125 mm × 100 mm
`43`	4:3	128 mm × 96 mm (default)
`32`	3:2	135 mm × 90 mm

21.5.4 Frame Shrinking

Even after adjusting frame size globally as discussed in §21.5.3, if any frame still fails marginally to display its entire contents, either the **squeeze** or **shrink** option may be used for shrinking a frame locally by a small amount.

The **squeeze** option reduces the vertical space between the enumerated and itemized items (i.e., **\item** in the **enumerate** and **itemize** environments) to zero. On the other hand, the **shrink** option shrinks the texts of a frame by the specified percentage. For example, a frame may be created as **\frame[squeeze]{}** or **\begin{frame}[squeeze]** for eliminating vertical space between enumerated and itemized items, or as **\frame[shrink=5]{}** or **\begin{frame}[shrink=5]** for shrinking all texts of the frame by 5%.

21.5.5 Removal of Headline/Footline and Sidebar

Headlines/footlines and sidebars are provided in slides for various reasons, such as displaying the status of a talk at any point of time, information about the presentation or author, containing navigational bars with clickable links to other frames, etc. However, they usually occupy a considerable amount of space of a slide, thus making it difficult in some cases to present a big piece of information, like a figure or a table, which can neither be accommodated in a single slide nor can be split over slides. As stated in §21.4.1, although the **compress** option in **\documentclass[ ]{beamer}** can reduce the space occupied by navigation bars in headline/footline and sidebar, it cannot remove them. Moreover, it is a global option to act on all the frames. Hence, in order to create a bigger space locally in a frame, its headline/footline or sidebar can be removed completely using the **plain** option in the frame. For example, a frame created as **\frame[plain]{}** or **\begin{frame}[plain]** will completely omit the headline/footline and sidebar of the frame, even if any theme (refer §21.4 for theme) is used with global effect for creating headline/footline and sidebar.

21.5.6 Frame Breaking

There may be cases where the entire contents of a frame cannot be displayed on a single slide (i.e., only a portion is displayed by truncating the remaining), even after changing the frame size, or shrinking a frame or removing its headline/footline and sidebar as discussed in §21.5.3–§21.5.5, respectively. Moreover, it also may not be very clear where to manually split such contents for putting in multiple frames, such as listed items, bibliographic list, or long equation array.

In above cases, the **allowframebreaks** option may be used in the frame (i.e., as **\frame[allowframebreaks]{}** or **\begin{frame}[allowframebreaks]**), which instructs to display the entire textual contents of a frame by allowing to break it into multiple slides, if required. In that case, the slides will be numbered by appending the frame title as `I`, `II`,…, e.g., `Advantages I`, `Advantages II`, etc. Further, the **allowdisplaybreaks** option (i.e., as **\frame[allowframebreaks, allowdisplaybreaks]{}** or **\begin{frame}[allowframebreaks, allowdisplaybreaks]**) may also be used if a long array of equations is to be split over multiple slides.

Hour 22

Slide Preparation II

Preparation procedures of simple slides under different themes are discussed in Hour 21. This Hour is devoted on some more important topics on slide preparation, such as piece-wise presentation of items, BEAMER environments, hyperlinking, etc.

22.1 Piece-Wise Presentation (BEAMER Overlays)

It is always preferred to present (cover, uncover/show, or highlight) the contents of a frame piece-wise, instead of displaying the entire contents in one go only as in the cases discussed in Hour 22. There are many processes for preparing frames, so as to present the contents piece-wise sequentially or in other orders. If the contents of a frame are to be presented piece-wise, the BEAMER will automatically split the frame into that number of slides.

22.1.1 Table of Contents

If a presentation is prepared section-wise, the table of contents of the presentation can be generated using the **\tableofcontents** command. Moreover, unlike in standard LaTeX, the BEAMER class also allows **\tableofcontents** to take an optional argument for creating certain special effects through various options. The very first such option is **pausesections** or **pausesubsections** (only one of them is to be used in a presentation). The **pausesections** option, to be used as **\tableofcontents[pausesections]**, internally issues a **\pause** command (§22.1.2 discusses **\pause** in detail) before each section entry in the table of contents, so as to uncover them piece-wise sequentially. The **pausesubsections** option does the same thing but in the case of subsections.

The second useful option is **currentsection**. It uncovers only the current section and its subsections under the table of contents, making it useful before starting the presentation of a new section. To get the effect, the table of contents is to be

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_22

repeated in an additional frame, say as \frame[t]{\tableofcontents[currentsection]}, immediately after each \section[]{} command. Similar to currentsection, BEAMER defines the currentsubsection option also, which uncovers only the current subsection in the current section under the table of contents. To get this effect, the table of contents is to be repeated in an additional frame, say as \frame[t]{\tableofcontents[currentsubsection]}, immediately after each \subsection[]{} command.

22.1.2 Uncovering Sequentially Using the \pause Command

The easiest way to uncover the contents of a frame is to use the \pause command, which pauses the display of the remaining contents/slides of the frame once a \pause is encountered. If it is inserted in multiple places in a frame, the contents of the frame will be uncovered only up to the first \pause in the first slide, up to the second \pause in the second slide, up to the third \pause in the third slide, and so forth, i.e., the effect of a \pause is terminated upon reaching at the next \pause or \onslide (§22.1.4 on the next page discusses \onslide) or the end of the frame.

An illustrative frame with the use of \pause is shown in Table 22.1 (output is

Table 22.1 Uncovering slide contents piece-wise using the \pause command

```
\begin{frame}[t]
  \frametitle{Seasons over the year} \pause
    \begin{enumerate}
    \item Summer \pause
    \item Autumn \pause
    \item Winter \pause
    \item Spring.
    \end{enumerate}
\end{frame}
```

not shown). Since \pause is inserted four times, the compilation will automatically split the frame over five slides as follows: the first slide to uncover only \frametitle{}, the second slide to uncover up to the first \item, the third slide to uncover up to the second \item, the fourth slide to uncover up to the third \item, and the fifth slide to uncover up to the fourth \item, which is also the end of the frame.

22.1.3 Uncovering Sequentially Using the Incremental Specification <+->

As stated in §22.1.2, the \pause command can be used anywhere for pausing the display of further contents/slides of a frame. However, if all the items of a listing environment (such as enumerate or itemize) are to be uncovered sequentially, instead of **\pause** at the end of each **\item**, the optional incremental specification **<+->**[1] may be used only once at the starting of the environment as shown in the left column of Table 22.2.

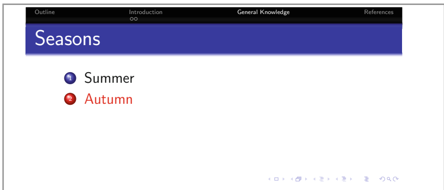
Table 22.2 Uncovering items piece-wise using the incremental specification **<+->**

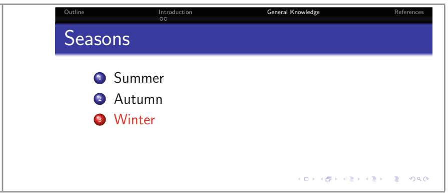
\begin{frame}[t] \frametitle{Seasons} \begin{enumerate}[<+->] \item Summer \item Autumn \item Winter \item Spring. \end{enumerate} \end{frame}	\begin{frame}[<+->][t] \frametitle{Animals} \begin{itemize} \item Cow \item Goat \end{itemize} % \begin{itemize} \item Lion \item Tiger \end{itemize} \end{frame}	\begin{frame}[t] \frametitle{Seasons} \begin{enumerate}[<+-|alert@+>] \item Summer \item Autumn \item Winter \item Spring. \end{enumerate} \end{frame}

If a frame contains multiple number of listing environments and their all items are to be uncovered sequentially, instead of inserting the optional incremental specification **<+->** in each environment, the same may be inserted only once in the frame as a whole as shown in the second column of Table 22.2. This is applicable not only to listed items, but also to other blocks, like theorem, proof, example, etc.

Further, as shown in the third column of Table 22.2, the incremental specification can be modified as **<+-|alert@+>** to alert (i.e., highlight) the current item by red color upon uncovering it. In this case, once the next item is uncovered, it will be highlighted quitting that of the previous item. The second and third slides of this frame under the **Frankfurt** presentation theme (refer §21.4.1 on page 209 for detail) is shown in Table 22.3.

Table 22.3 Slides with incremental overlay-specification under the **Frankfurt** presentation theme for the input frame of the third column of Table 22.2





*22.1.4 Other Piece-Wise Presentation Specifications**

When it is sought to present (i.e., cover, uncover, or highlight) the contents of a frame piece-wise, the BEAMER automatically splits the frame into that number of slides. A piece-wise presentation (overlay) specification in **<>** specifies the slide numbers

[1]Specific slides of a frame in which a particular item or block is to be shown/not shown are specified in **<>**, which in the BEAMER class is known as the overlay specification.

of a frame in which a particular component is to be presented. The rules of such specifications are given in Table 22.4 with some examples. While specifying a slide number in an overlay specification, care must be taken that the corresponding frame has been split at least up to that number of slides, otherwise the intermediate gap will be filled up by generating some identical slides (i.e., slides with the same contents) in between.

Table 22.4 Rules for piece-wise presentation (overlay) specification

Specification	Meaning
<3>	Slide 3 only
<1,2,4>	Slides 1, 2 and 4 only
<3-6>	Slides 3–6 (i.e, slides 3, 4, 5, 6) only
<3->	Slide 3 onward all sides of the frame
<-4>	All starting slides up to slide 4 (i.e., slides 1–4) only
<2,4-6,8,11->	Slides 2, 4–6, 8, 11- (i.e., slides 2, 4, 5, 6, 8, 11 and rest of the slides of the frame).

There are many commands which can take overlay specifications, as shown in Table 22.4, to instruct in which slides their contents are to be presented. Such commands[2] are listed in Table 22.5 with some explanatory texts stating the functions

Table 22.5 Commands which can take overlay specifications

Command	Example stating the function
\textbf<>{}	\textbf<3>{It is boldfaced in slide 3, and in normal fonts in all other slides}
\textit<>{}	\textit<4>{It is in italic fonts in slide 4, and in normal fonts in others}
\textrm<>{}	\textrm<5>{It is in serif fonts in slide 5, and in normal fonts in others}
\textsf<>{}	\textsf<6>{It is in sans serif fonts in slide 6, and in normal fonts in others}
\textsl<>{}	\textsl<7>{It is in slanted shape in slide 7, and in normal fonts in others}
\alert<>{}	\alert<1>{It is shown in red color in slide 1, and in normal color in others}
\color<>[]{}{}	\color<2>[rgb]{0,0,1}{It is in blue color in slide 2, and in normal color in others}
\only<>{}	\only<1>{It is shown in slide 1 only, and the space is freed in others}
\onslide<>{}	\onslide<2>{It is shown in slide 2 only, and the space is kept blank in others}
\uncover<>{}	\uncover<3>{It is shown in slide 3 only, and kept covered or transparent in others}
\visible<>{}	\visible<4>{It is shown in slide 4 only, and the space is kept blank in others}
\invisible<>{}	\invisible<5>{It is not shown in slide 5 only, but the space is kept blank}
\alt<>{}{}	\alt<6>{It is shown in slide 6 only}{It is shown in all slides other than in 6}
\temporal<>{}{}{}	\temporal<7>{It is shown in slides 1–6}{It is shown in slide 7 only}{It is shown in slides 8-}
\item<>	\item<8> It is shown in slide 8 only, and the space is kept blank in others.

of the commands. Although only one slide number is specified in each command, they (except \alt<>{}{}) can take any type of specifications as shown in Table 22.4. The \alt<>{}{} command can take only one slide, in which its first argument will be shown and the second argument will be shown in all other slides of a frame. On the other hand, the \temporal<>{}{}{} command alternatively handles three arguments.

[2] In the case of a command having arguments, the overlay specification is put in <> in between the command and its arguments (refer Table 22.5 for examples).

The first argument is shown in all slides prior to the specified ones, the second argument is shown in the specified slides only, and the third argument is shown in all the slides appearing after the specified ones. If discontinuous slides are specified in `\temporal<>{}{}{}` (say, 2,4,7), the intermediate non-specified slides are also treated to be prior to the specified ones and hence the first argument will be shown in those non-specified intermediate slides. Note that the `alert@` command, used in the third column of Table 22.2 for alerting/highlighting an item, can also be used in the commands of Table 22.5 for alerting an item in particular slides, e.g., `\item<2-|alert@2>` for alerting the item in slide 2, or `\item<alert@2,4>` for alerting the item in slides 2 and 4.

The overlay-specification supported commands shown in Table 22.5 can be used independently as well as in combination. Moreover, different commands can be used for similar effect, which is clear from their functions given in Table 22.5. Such applications through two frames under the **Hannover** presentation theme (refer §21.4.1 for detail) are shown in Table 22.6 (refer Table 21.1 for detail coding). The first frame

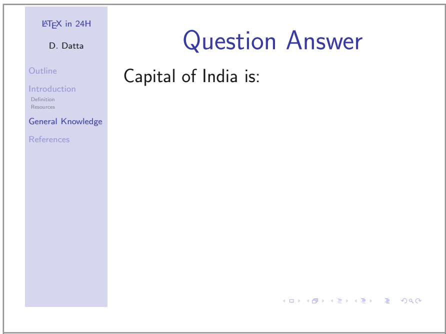
Table 22.6 Applications of some overlay-specification supported commands

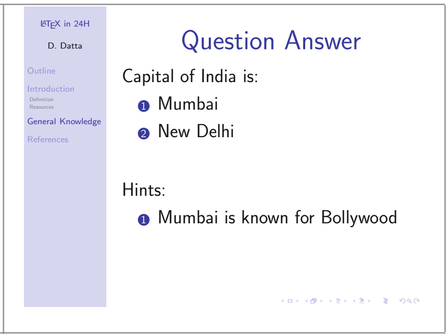
```
\begin{frame}[t]
    \uncover<1-2>{Result: $x+y$ =}
    \alt<1>{\textbf{?}\quad guess what}{\textbf{$z$}\quad\checkmark}
\end{frame}
%
\begin{frame}[t]
    \frametitle{Question Answer}
    \uncover<1->{Capital of India is:}
    \begin{enumerate}
    \item<2-> Mumbai
    \item<4-> \color<6>[rgb]{0,0,1}{New Delhi}
    \end{enumerate}
    \vskip 10mm
    \only<3-5>{Hints:}
    \begin{enumerate}
    \item<3-5> Mumbai is known for Bollywood
    \item<5> Parliament House is in New Delhi
    \end{enumerate}
\end{frame}
```

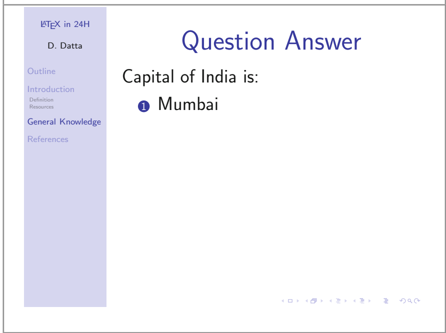
will generate two slides, printing "`Result:` $x+y=$ **?** `guess what`" in the first slide and "`Result:` $x+y=\mathbf{z}$ ✓" in the second slide. On the other hand, the second frame will generate six slides, which are shown column-wise in Table 22.7 on the next page. Notice in these two frames that the slides, in which the items will be presented, are not specified in order. For example, in the second frame, the second `\item` of the first `enumerate` environment will be shown in slides 4–6, whereas the `\item` of the second `enumerate` environment will be shown in the earlier slide 3 and intermediate slide 5.

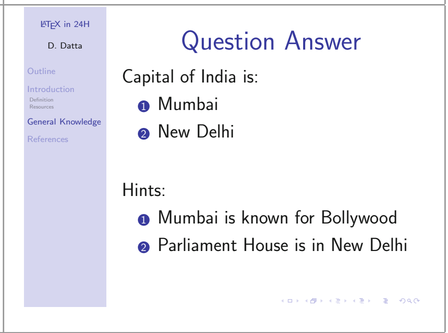
However, even if the presenting slides are not specified in order in an input file, the positions of the items in any output slide will be in the same order in which they are inserted in the input file[3], which can be noticed in Table 22.7.

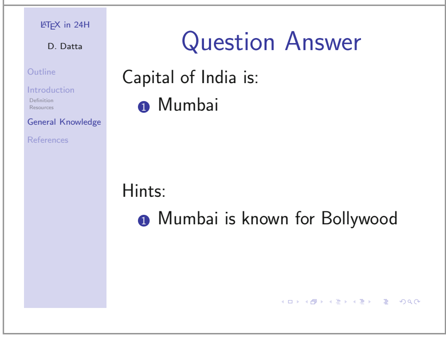
Table 22.7 Slides with overlay specification under the **Hannover** presentation theme for the second input frame of Table 22.6

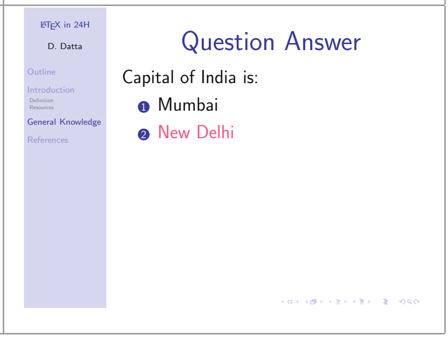












[3] Even if the presenting slides are not specified in order in overlay specifications, the positions of the items in any output slide will be in the same order in which they are inserted in the input file.

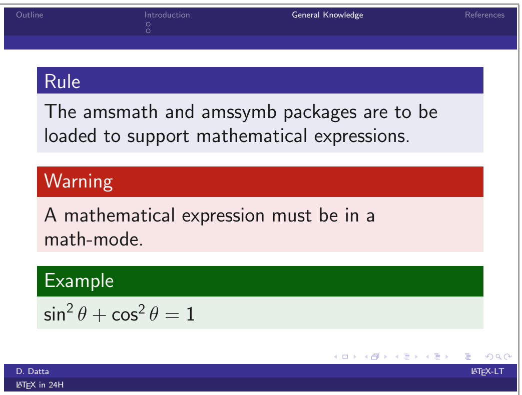
22.2 Environments in BEAMER Class*

The BEAMER class defines the **visibleenv**, **invisibleenv** and **uncoverenv** environments, whose effects are the same, respectively, with those of the **\visible{}**, **\invisible{}** and **\uncover{}** commands as stated in Table 22.5.

The BEAMER class also defines some block-type environments for producing a piece of texts with a user-defined heading. Such environments include **block**, **alertblock** and **exampleblock**, which are generally meant for a normal block of texts, an alerting message, and an example-like illustration, respectively. The **beamerboxesrounded** is another block-type environment, whose contents are framed by a rectangular area with rounded corners. A block-type environment takes its heading as an argument, and it is highlighted in the **alertblock** environment. A block-type environment also has the provision for taking overlay specifications, on either side of the heading argument, so as to uncover the environment in the specified slides only. Applications of the block-type environments under the **Berlin** presentation theme (refer §21.4.1 for detail) are shown in Table 22.8 (refer Table 21.1 on page 206 for detail coding). In this case the **\setbeamertemplate{blocks}[rounded][shadow=true]** command is also inserted in the preamble, where the argument **blocks** is used for defining the style of the background blocks of the environments as follows: the

Table 22.8 Slides with block-type environments under the **Berlin** presentation theme

```
\begin{frame}[t]
\begin{block}{Rule}
The amsmath and amssymb ...
\end{block}
%
\begin{alertblock}<2->{Warning}
A mathematical expression ...
\end{alertblock}
%
\begin{exampleblock}{Example}<3>
$\sin^2\theta + \cos^2\theta = 1$
\end{exampleblock}
\end{frame}
```

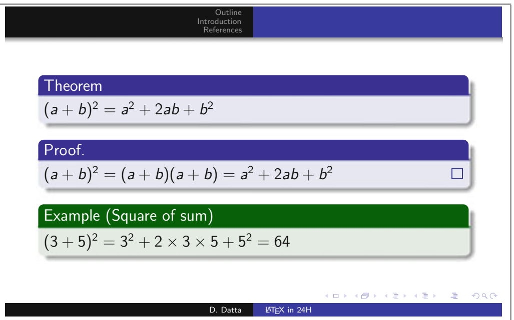


rounded option instructs to round off the corners of the blocks (if not by default), and the **shadow=true** option allows to draw shadows behind the blocks as clearly visible in the output slide shown in Table 22.8. In regard of the frame, it is split over three slides as per the used overlay specifications (only the last slide is shown in Table 22.8). Since the **block** environment will be shown in all the slides of the frame, no overlay specification is required to it. Also notice in the **alertblock** and **exampleblock** environments that an overlay specification can be inserted on either side of the heading argument of a block-type environment.

Apart from the above, the BEAMER class defines some theorem-like environments for producing a piece of texts with a default heading. Such environments include **corollary**, **definition**, **definitions**, **example**, **examples**, **fact**, **proof**, and **theorem**. The theorem-like environments work in a similar way with those of block-type environments, except that they do not need a mandatory heading argument but an optional argument may be provided as an additional heading. The additional heading is generally produced in a pair of parentheses after the default heading, except in the **proof** environment. In the **proof** environment, the default heading "`Proof.`" is replaced by the optional heading. Moreover, by default a **proof** environment is ended by a right aligned **\qed** symbol (a small square). Applications of the **theorem**, **proof** and **example** environments under the **Warsaw** presentation theme (refer §21.4.1 for detail) are shown in Table 22.9 (refer Table 21.1 on page 206 for detail coding). In regard of the frame, it is split over three slides as per the used overlay specifications (only the last slide is shown in Table 22.9). Since the **theorem** environment will be shown in all the slides of the frame, no overlay specification is required to it. Moreover, an optional heading is provided to the **example** environment, which is produced in a pair of parentheses after the default heading.

Table 22.9 Slides with theorem-like environments under the **Warsaw** presentation theme

```
\begin{frame}[t]
\begin{theorem}
$(a+b)^2 = a^2 + 2ab + b^2$
\end{theorem}
%
\begin{proof}<2->
$(a+b)^2=(a+b)(a+b)=a^2+2ab+b^2$
\end{proof}
%
\begin{example}<3->[Square of sum]
$(3+5)^2=3^2+2\times3\times5+5^2=64$
\end{example}
\end{frame}
```



22.3 Table and Figure in Presentation*

Tables in BEAMER presentation also can be prepared using the standard LaTeX environment **table** without any option for vertical positioning. Inside the **table** environment (even without the **table** environment), either **tabular** or **tabularx** environment can be used for generating tabular cells. Two such examples in a single frame are shown in Table 22.10 on the next page. The used overlay specifications split the frame over six slides. The first example is displayed in the first two slides and the second example in the next four slides. In the first example, the entire table is displayed in slide 2 as a whole, as it is inserted in **\only<2>**. Further, the table is put in **\color<2>[ ]{ }{ }** for producing it in given color. On the other hand, the table of the second example is inserted in **\onslide<4->** for displaying it in all slides starting from slide 4, in such a way that the first row (after the heading row) will be uncovered in slides 5 and 6,

Table 22.10 Table in BEAMER presentation

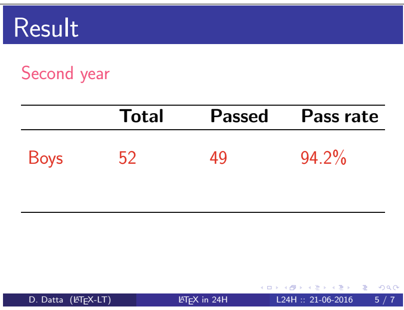
```
\begin{frame}[t]
   \frametitle{Result}
   % Example 1
   \only<1-2> {\color<1-2>[rgb]{1,0.3,0.5}{First year}}
   \only<2> {\color<2>[rgb]{1,0.3,0.5}{
      \begin{table}
      \flushleft
      \begin{tabular}{cccc}
        \hline         & {\bf Total}  & {\bf Passed}  & {\bf Pass rate}\\
        \hline  Boys   &    56        &    50         &    89.3\%\\
                Girls  &    38        &    36         &    94.7\%\\
        \hline
      \end{tabular}
      \end{table}  }}
   % Example 2
   \onslide<3-> {\color<3->[rgb]{1,0.3,0.5}{Second year}}
   \onslide<4-> {\color<4->[rgb]{0,0,0}{
      \begin{table}
      \begin{tabularx}{\linewidth}{XXXX}
      \hline   &   {\bf Total}&    {\bf Passed}&    {\bf Pass rate}\\
      \hline \uncover<5->{\alert<5>{Boys}}& \uncover<5->{\alert<5>{52}}&
                 \uncover<5->{\alert<5>{49}}& \uncover<5->{\alert<5>{94.2\%}}\\
             \uncover<6>{\alert<6>{Girls}}& \uncover<6>{\alert<6>{46}}&
                 \uncover<6>{\alert<6>{41}}& \uncover<6>{\alert<6>{89.1\%}}\\
      \hline
      \end{tabularx}
      \end{table}  }}
\end{frame}
```

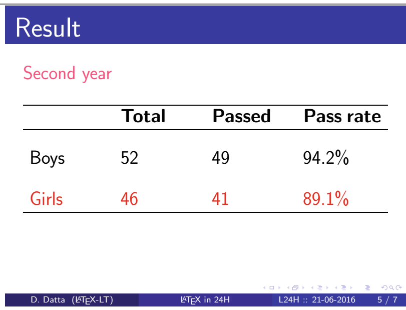
while the second row in slide 6 only (these two slides under the **Madrid** presentation theme are shown in Table 22.11 on the next page). Since more than one cell of a table cannot be presented through a single overlay specification[4], the entry of each cell is inserted in separate **\uncover<>{}**, also in **\alert<>{}** for highlighting them in the specified slides. Note that apart from inserting in **\onslide<4->{}**, the table is also inserted in **\color<4->[rgb]{0,0,0}{}** for producing it in black color, otherwise it would have been produced in the same color specified for the previous slide, i.e., for slide 3.

Like tables, figures from external files also can be inserted in BEAMER presentation using the standard LaTeX environment **figure** without any option for vertical positioning. Inside the **figure** environment (even without the **figure** environment), a figure can be inserted through the standard LaTeX commands **\epsfig{}** and **\includegraphics[]{}**. The **\includegraphics[]{}** command can take overlay specifications also, in which case its form becomes **\includegraphics<>[]{}**, where overlay specifications as stated in Table 22.4 are inserted in **<>**. For example,

[4]More than one cell of a table cannot be presented through a single overlay specification, but each one through a separate specification.

Table 22.11 Table in slides with overlay specification under the **Madrid** presentation theme for the second example of Table 22.10





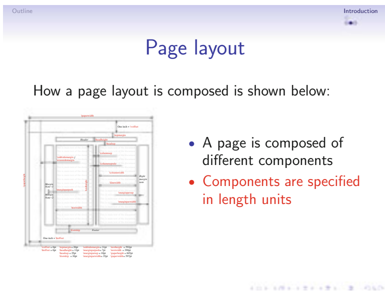
\includegraphics<2>[`width=5`**cm]{**`tiger`**}** for displaying the specified figure in slide 2 only, or **\includegraphics<**`2,5`**>[**`width=5`**cm]{**`tiger`**}** for displaying it in slides 2 and 5.

22.4 Dividing a Frame Column-Wise*

It is often required to present some materials in a frame side-by-side, e.g., showing a figure on the left side and explaining it on the right side. As the example shown in Table 22.12 along with a output slide under the **Singapore** presentation theme,

Table 22.12 Slides with side-by-side materials through the **columns** environment under the **Singapore** presentation theme

```
\begin{frame}
\frametitle{Page layout}
How a page … below:\pause\vskip 5mm
\begin{columns}
\column{0.4\textwidth}
\includegraphics[width=\textwidth]{layout}
%
\column{0.6\textwidth}
\begin{itemize}[<+-|alert@+>]
\item A page is composed of different …
\item Components are specified in length …
\item Length of a component can be …
\end{itemize}
\end{columns}
\end{frame}
```



such a frame can be produced through the **columns** environment, inside of which each column is created using a **\column{**`cwidth`**}** command with `cwidth` being the horizontal width of the column.

22.5 Repeating Slides in Presentation*

During a presentation, sometime it is required to go back to a particular slide of a previous frame for explanation purpose. The BEAMER class provides the **label** option and **\againframe<>{}** command for this purpose, avoiding the need to scroll back to the required slide. In this case, the previous frame is to be labeled with the **label** option, say as **\begin{frame}[label=**`stress`**]** to label the frame by `stress`. Then the **\againframe<>{}** command is to be inserted in the required location for reproducing the particular slide of the frame, say as **\againframe<**`3`**>{**`stress`**}** for reproducing slide 3 of the frame which is labeled by `stress`. Without any overlay specification, i.e., inserting as **\againframe{**`stress`**}**, the command will reproduce the entire frame (i.e., its all slides, if produced more than one using any overlay specification in the frame). Note that the **\againframe<>{}** command is to be inserted outside of any frame.

22.6 Jumping (Hyperlink) to Other Slides*

Provision is there in the BEAMER class to jump from the current slide (during presentation) to another slide for various purposes, such as referring to (showing) materials included in another slide, going back after referring, skipping some intermediate slides, etc. It can be done by first issuing a label-word to the frame containing the target slide where the jump is to be made (refer §22.5 for issuing a label-word), and then creating a hyperlinked button in the current slide to go to the target slide by clicking the hyperlinked button. Such a button is created through a button-type command with the texts, to be displayed in the button, as its argument. Then the button is activated (hyperlinked) by inserting it in a hyperlink-type command.

Some commonly used button-type commands are **\beamerbutton{}** (draws a simple button), **\beamergotobutton{}** (draws a button with a right-pointing arrow), **\beamerreturnbutton{}** (draws a button with a left-pointing arrow), and **\beamerskipbutton{}** (draws a button with a double right-pointing arrow). The argument of such a command is the name of the generated button (i.e., texts to be displayed in the button), e.g., `Go for detail`, `Return`, `Skip proof`, etc.

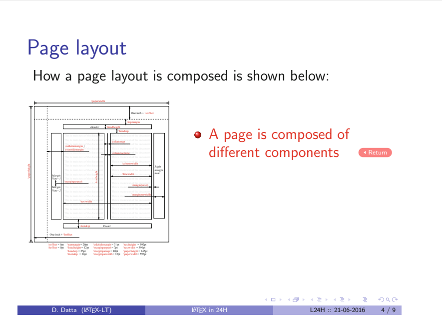
Similarly, the following are the commonly used hyperlink-type commands: **\hyperlink<>{}{}** (links the specified slide), **\hyperlinkframestart<>{}{}** (links the first slide of the current frame), **\hyperlinkframeend<>{}{}** (links the last slide of the current frame), **\hyperlinkframestartnext<>{}{}** (links the first slide of the next frame), **\hyperlinkframeendprev<>{}{}** (links the last slide of the previous frame), **\hyperlinkpresentationstart<>{}{}** (links the first slide of the presentation), and **\hyperlinkpresentationend<>{}{}** (links the last slide of the presentation). The overlay specification of these commands in **<>** is the slide in which the hyperlinked button is to be shown, the first mandatory argument in **{}** is the label-word of the target frame along with the overlay specification for the target slide in that frame (e.g., `stress` **<**`2`**>** for slide 2 of the frame having the label-word `stress`), and the second mandatory argument in **{}** is the button-type command as stated above. Note that, in the absence of an overlay specification in a hyperlink-type command, the hyperlinked button will be visible in all the slides of the frame in which it is inserted.

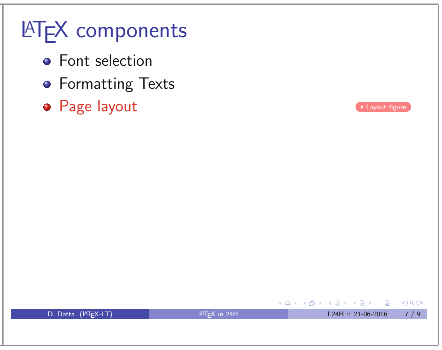
A hyperlinking example is shown in Table 22.13 along with its output in Table 22.14, where an earlier slide is hyperlinked from a latter slide, because of

Table 22.13 Hyperlinking slides in BEAMER presentation

```
\begin{frame}[t,label=layout]
  \frametitle{Page layout}
  How a page layout is composed is shown below:\pause \vskip 5mm
  \begin{columns}
    \column{0.4\textwidth}
    \includegraphics[width=\textwidth]{layoutpic}
    %
    \column{0.6\textwidth}
    \begin{itemize}[<+-|alert@+>]
    \item A page is composed of different components \hfill
                                 \hyperlink<2>{LaTeX<3>}{\beamerreturnbutton{Return}}
    \item Components are specified in length units
    \item Length of a component can be changed manually
    \end{itemize}
  \end{columns}
\end{frame}
%
\begin{frame}[t,label=LaTeX]
  \begin{itemize}[<+-|alert@+>]
  \frametitle{\LaTeX\ components}
  \item Font selection
  \item Formatting Texts
  \item Page layout \hfill      \hyperlink<3>{layout<2>}{\beamergotobutton{Layout figure}}
  \item Table, figure, equation, etc.
  \end{itemize}
\end{frame}
```

Table 22.14 Hyperlinked slides under the **Boadilla** presentation theme for the input frames of Table 22.13





which a return hyperlink is also added in the earlier slide (a return button seems essential since the presentation will not return automatically from the linked slide to the slide where it was linked). In this example, the combined command '**\hyperlink<**3**>{**layout**<**2**>}{\beamergotobutton{**Layout figure**}}**' is inserted to create a button with texts '`Layout figure`' in slide 3 of the frame `LaTeX`, so that a click on the button will lead the presentation to slide 2 of the frame `layout`. Similarly, the button with texts '`Return`', created using **\hyperlink<**2**>{**LaTeX**<**3**>}{\beamerreturnbutton{**Return**}}** in slide 3 of the frame `layout`, will return the presentation to slide 3 of the frame `LaTeX`.

Hour 23

Error and Warning Messages

Commitment of typographical or logical errors is unavoidable. Most of the errors and warnings in LaTeX can be debugged easily, while some could be a little bit complicated. Some commonly committed errors and subsequent error or warning messages generated by a command-line-based LaTeX compiler are discussed in this Hour.

During compilation of a LaTeX input file, a lot of process-related internal information are displayed, which are not so important particularly for beginners. If the input file, say named '`lsw.tex`', is free from any error that must be debugged, the compilation will be completed with the following two lines of final information:

```
Output written on lsw.dvi (12 pages, 50028 bytes).
Transcript written on lsw.log.
```

In the case of an erroneous file requiring mandatory debugging, however, the compilation will be paused showing the relevant error message immediately after arriving at the first error in the file. In most of the cases, the compilation will be paused with a '`?`' mark after the error message. By the `?` mark, the LaTeX compiler asks the user for an action, which can be responded as follows:

▷ If the error is not clear, '`h`' may be entered for help.

▷ Allow by pressing the `Enter` key to try to compile without debugging the error.

▷ If the error is clear, '`e`' may be entered to lead to the error in the input file. However, care must be taken with this option, since the compiler will open the input file in the terminal if it fails to communicate with the used LaTeX editor. In that case, editing the input file in the terminal, while the same is open in the LaTeX editor also, may even corrupt the input file.

▷ Alternatively, terminate the compilation with '`Ctrl+z`', and then edit the input file manually in the used LaTeX editor.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_23

23.1 Error Message

Immediately after detecting the first error that must be debugged, the compilation of the LaTeX input file will either be paused without generating any output, or be terminated by generating output up to the previous page of that containing the first error. Then a user needs to take necessary steps to debug the input file according to the error message displayed by the LaTeX compiler. Some commonly generated error messages and their possible reasons are presented below in alphabetical order.

Error: `*` (no error message, just paused).

✓ **\end{document}** is missing at the end of the LaTeX input file.

Error: `! Display math should end with $$.`

✓ A mathematical symbol or expression is inserted in **$ $** in a math-mode, such as in the **equation** environment or in the **\(\)**-mode. Following the error message, the serial number of the line in which the error occurs will also be shown marking with `l.`, along with the erroneous expression at the end of the line, e.g.,

```
l.349    x+y = $\alpha
```

Error: `! Double subscript.`

✓ Subscript is inserted twice to a single character or symbol, e.g.,

```
l.54    \frac{x_i_j}{y}
```

Error: `! Double superscript.`

✓ Superscript is inserted twice to a single character or symbol, e.g.,

```
l.54    \frac{x}{y^t^2}
```

Error: `! LaTeX Error: Environment --- undefined.`

✓ A non-existing or mis-spelled environment is used, e.g.,

```
l.54    \begin{enumerating}
```

✓ Environment is correct, but its supporting package is not loaded. LaTeX will not say anything about the package, but the environment will be shown undefined, e.g.,

```
l.211    \begin{wrapfigure}
```

In the above example, **wrapfigure** is a correct environment, but the error is shown since its supporting package **wrapfig** is not loaded in the preamble.

Error: `! Extra }, or forgotten \endgroup.`

✓ Something is ended by '**}**' without starting it with '**{**', e.g.,

```
l.7    ... as follows for a file named \tt lsw.tex}
```

✓ An environment is used without starting it with **\begin{}**. Following the error message, the ending line of the environment will be shown.

Error: `! Extra alignment tab has been changed to \cr.`

✓ Extra entries with **&** are inserted in a row of a column-based environment, like **array**, **eqnarray**, **tabular**, or **tabularx**. This may also happen if two rows are not separated by ****, or if a column is generated with a wrong option, e.g., use of **X** in the **tabular** environment (**X** is permitted in the **tabularx** environment only). The erroneous row or the ending line of the environment will be shown after the error message.

✓ Non-existing column number is used in **\cline{}** in the **tabular** or **tabularx** environment.

Error: `! Extra \right.`

✓ Delimiter height adjustment command **\right** is used without preceding by its complementary command **\left**, e.g.,

```
l.35    \frac{x}{y} \right)
```

Error: `! File ended while scanning use of \---.`

✓ See `Runaway argument?`

Error: `! Illegal parameter number in definition of \---.`

✓ More number of arguments is used in **\newcommand{}{}** than defined, e.g.,

```
l.48    \newcommand{\sq}[1]{#1^{#2}}
```

Error: `! Illegal unit of measure (pt inserted).`

✓ Unrecognized length unit is used, e.g.,

```
l.24    \begin{minipage}[t]{5.0cn}
```

Error: `! Improper \prevdepth.`

✓ Previous environment is not closed by **\end{}**. The ending line of the next environment will be shown after the error message.

Error: `! LaTeX Error: \begin{nenv} on input line -- ended by \end{document}.`

✓ Environment 'nenv' is opened with **\begin{**nenv**}** in line `--`, but the compilation reached the end of the input file, i.e., **\end{document}** command, before it is closed with **\end{**nenv**}**.

Error: `! LaTeX Error: \begin{nenvs} on input line -- ended by \end{nenvf}.`

✓ Environment `nenvs` is opened with **\begin{**`nenvs`**}** in line `--`, but forgotten to close it by **\end{**`nenvs`**}** (or mistakenly closed by **\end{**`nenvf`**}**). The ending line of the next environment (or the erroneous closing line) will be shown after the error message.

Error: `! LaTeX Error: Can be used only in preamble.`

✓ Something is inserted after **\begin{document}**, which is permitted in the preamble only, e.g., **\usepackage{}**.

Error: `! LaTeX Error: \caption outside float.`

✓ The **\caption{}** command is inserted outside a float environment, like **table** or **figure**.

Error: `! LaTeX Error: Command \--- already defined.`

✓ Attempt is made to define an existing command as a new command, which is not permitted, e.g.,

```
l.29    \newcommand{\alpha}{A Greek letter}
```

Error: `! LaTeX Error: File '---' not found.`

✓ The figure file '`---`' inserted through **\epsfig{}** or **includegraphics[]{}** is either not available or not in proper format, e.g.,

```
l.246    ...ludegraphics{file=pot.eps,width=2cm}
```

In the above example, the figure file might be available, but it is read as '`file=pot.eps,width=2cm`', not as '`pot.eps`'. This is due to the fact that the figure is inserted using the syntax of **\epsfig{}** in **\includegraphics[]{}**.

Error: `! LaTeX Error: File '---.cls' not found.`

✓ Either an unavailable or a mis-spelled document-class file is inserted in **\documentclass{}** (`cls` extension means a document-class file).

Error: `! LaTeX Error: File '---.sty' not found.`

✓ Either an unavailable or a mis-spelled package file is loaded through **\usepackage{}** in the preamble (`sty` extension means a style file, which is used as a package).

Error: `LaTeX Error:Lonely \item--perhaps a missing list environment.`

✓ The **\item** command is inserted outside a listing environment, like **enumerate**, **itemize**, or **description**. The erroneous line, or the next line if the **\item** command has no content, will be shown after the error message.

Error: `! LaTeX Error: Missing \begin{document}`

✓ Either some contents of a document are inserted before **\begin{document}** or **\begin{document}** is missing.

Error: `! LaTeX Error: No \title given.`

✓ The **\maketitle** command is not preceded by **\title{}**, or it is missing.

Error: `! LaTeX Error: Not in outer par mode.`

✓ Two non-permitted environments are nested, like **figure** in **minipage** or **table**. Following the error message, the first line in the inner environment will be shown.

Error: `! LaTeX Error: Option clash for package ---.`

✓ The same package is loaded multiple times using **\usepackage[]{}** with different options. The next line after the last erroneous package loading will be shown after the error message.

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

✓ No **\item** is inserted in a listing environment, like **enumerate**, **itemize**, or **description**. The ending line of the environment will be shown after the error message.

✓ A non-permitted environment is nested inside a secondary environment, like **verbatim** in **tabular**.

Error: `! LaTeX Error: There's no line here to end`

✓ The **\vspace{}** command is preceded by a blank line, or followed by a new line or a line break command, like ****, **\newline** or **\linebreak**. Following the error message, the next line that of **\vspace{}** will be shown.

Error: `! LaTeX Error: Too deeply nested.`

✓ Excess number of environments are nested. The starting line of the first excess nested environment will be shown after the error message.

Error: `! LaTeX Error: Too many columns in eqnarray environment.`

✓ More than three columns, i.e., more than two **&**, are used in a single row of the **eqnarray** environment. The ending line of the environment will be shown after the error message.

Error: `! LaTeX Error: Undefined tab position.`

✓ A tab is used in a row of the **tabbing** environment, which is not defined earlier, e.g.,

`l.253    Volume (V)    \> = bdh    \> High`

Error: `! LaTeX Error: Unknown option '---' for package '---'.`

✓ An unknown option is inserted in **\usepackage[]{}** while loading a particular package. The erroneous line, shown after the error message, may be somewhat strange, e.g.,

`l.40    \RequirePackage`

Error: `! LaTeX Error: \verb illegal in command argument.`

✓ The **\verb" "** command is inserted in the argument of another command, e.g.,

`l.54    \textcolor{red}{\verb"This is in red color"}`

Error: `! Misplaced alignment tab character &.`

✓ Alignment tab **&** is inserted in a wrong place (it is used in column-based environments, like **array**, **eqnarray**, **tabular**, or **tabularx**).

Error: `! Misplaced \noalign.`

✓ Previous row prior to a **\hline** command in the **tabular** or **tabularx** environment is not ended by the line-break command ****. The ending line of the environment will be shown after the error message.

Error: `! Missing } inserted.`

✓ Something is started with '**{**' in a paragraph within an environment other than **document**, but it is not closed by '**}**'. The ending line of the environment will be shown after the error message. Since the exact location of the error is not shown, it may take some time to manually locate the error inside the environment.

Error: `! Missing $ inserted.`

✓ Either the symbol or the environment shown after the error message is to be inserted in a math-mode (say in **$ $**, **\(\)**, **\[\]**, or **equation** environment), e.g.,

`l.304    Greek letter \alpha`

or,

`l.364    \begin{array}{ll}`

Error: `! Missing control sequence inserted.`

✓ The first argument of **\newcommand{}{}** or **\renewcommand{}{}** is not a command, i.e., not preceded by a \, e.g.,

`l.42    \newcommand{sq}[2]{#1^{#2}}`

Error: `! Missing \endcsname inserted.`

✓ The name of an environment is preceded by a **** in **\begin{}** or **\end{}**, e.g.,

`l.24    \begin{\tabular}`

✓ The first argument of **\newcommand{}{}** is missing. The next line of the erroneous one will be shown after the error message.

Error: `! Missing \endgroup inserted.`

✓ In a group of nested environments, an inner environment is not ended by **\end{}**, e.g., while using **tabular** in **table** and closing **table** by **\end{table}** without closing **tabular** by **\end{tabular}**.

Error: `! Missing number, treated as zero.`

✓ A numerical-valued argument of a command or environment is missing, or a nonnumeric value is assigned to it. The erroneous line, or the next line if the argument was the last term of the erroneous line, will be shown after the error message, e.g.,

```
l.34    \hspace c
```

or,

```
l.25    \begin{tabularx}{}{|l|c|c|}
```

Error: `! Missing \right. inserted.`

✓ Delimiter height adjustment command **\left** is not closed by the complementary command **\right**. Following the error message, the erroneous line or the ending line of the environment in which the error occurs will be shown, e.g.,

```
l.116    \[f\left(\frac{x+1}{2}+2)\]
```

Error: `! Package varioref Error: \vref at page boundary -- - -- (may loop).`

✓ The effect of the **\vref{}** command falls in the last line of a page of the output file, which will be displayed as, e.g.,

```
l.630    ...ed in \S\vref{sec:colwidth}
```

This error is something like a warning only. Hence, the compilation may be allowed to continue without bothering about it.

Error: `! Package xcolor Error: Undefined color '---'.`

✓ The inserted color is not defined through **\definecolor{}{}{}** supported by the **color** package. Following the error message, the erroneous line will be shown, e.g.,

```
l.263    ...r{LightBlue}{This is in light blue color}
```

Error: `! Paragraph ended before \--- was complete.`

✓ See `Runaway argument?`

Error: `Runaway argument?`

✓ Something is started with '**{**', but not closed by '**}**'. The line number shown below the error message may point to a broader location of the error. However, the exact location of '**{**' can be found easily by checking the immediate next line of the error message and then it can be closed by '**}**' in the proper location, e.g.,

```
{enumerate \section {Unnumbered listing} \ETC.
```

or,

```
Column {\bf X is used in tabular environment \ETC.
```

The above two examples indicate that '**{**' prior to '`enumerate`' and '`\bf`', respectively, are not closed by '**}**'.

Error: `! TeX capacity exceeded, sorry [input stack size=---].`

✓ The first argument of **\renewcommand{}{}** is empty, or a fragile command is inserted as an argument of another command, say **\footnote{}** in **\section{}**. Compilation will not be paused in any of the cases, but the output file will be generated up to the previous page of that containing such error. Following the error message, the erroneous line will be shown, e.g., `l.32    \renewcommand{}`

or,

`l.213    ...{\footnote{} is a fragile command.}`

Error: `! Too many }'s`

✓ See `! Extra }, or forgotten \endgroup.`

Error: `! Undefined control sequence.`

✓ An unavailable or mis-spelled command is inserted, e.g.,

`l.60  ...ommand is inserted. The Greek letter \alfa`

✓ Command is correct, but its supporting package is not loaded. LaTeX will not say anything about the package, but the command will be shown as an undefined control sequence, e.g.,

`l.78    This is in \textcolor`

In the above example, **\textcolor** is a correct command, but the error is shown since its supporting package **color** is not loaded in the preamble.

23.2 Warning Message

In the case of not having any error that must be debugged, the compilation of a LaTeX input file, say named '`lsw.tex`', by the `latex` command will generate the output with the following two lines of final information:

```
Output written on lsw.dvi (12 pages, 50028 bytes).
Transcript written on lsw.log.
```

If the input file contains any labeling and cross-referencing, i.e., labeling numbered items with **\label{}** and referring them with **\ref{}**, or citing a bibliographic item with **\cite{}**, two or three lines of warning messages may also be seen somewhere above the two lines of final information mentioned above. Such warning messages will be like those as shown below:

```
... Warning: There were undefined citations.
LaTeX Warning: There were undefined references.
LaTeX Warning: There were multiply-defined labels.
```

The first two warning messages may disappear upon completing the compilation of the input file as discussed in §14.3, 15.4 or 16.2.3. However, it is a matter of concern if any of such warning messages still remains.

Warning: ... `Warning: There were undefined citations.`

✓ This is a warning from the bibliographic reference generator, e.g., '`Package natbib Warning`' if bibliographic references are generated through the **natbib** package. The warning message means that some citation keys used in the **\cite{}** command are unknown, may be undefined or mis-spelled in the bibliographic list. Checking the list of information generated by the LaTeX compiler, such errors can be found very easily, e.g.,

```
Package natbib Warning: Citation 'lko'
  on page 8 undefined on input line 271.
```

✓ Alternatively, the output file (i.e., the `.dvi` file) may be checked for '`?`' sign, where each of such errors will be marked by a '`?`' sign. Once the location of an error is found in the `.dvi` file, the required correction can be made in the input file.

Warning: `LaTeX Warning: There were undefined references.`

✓ This warning message says that some reference keys used in the **\ref{}** command (or other similar commands, such as **\vref{}**, **\pageref{}**, **\vpageref{}** and **\eqref{}**) are either mis-spelled or not defined through the **\label{}** command. As in above, these errors can also be found very easily by checking the list of information generated by the LaTeX compiler, e.g.,

```
LaTeX Warning: Reference 'pichart'
  on page 8 undefined on input line 269.
```

✓ Alternatively, the output file (i.e., the `.dvi` file) may be checked for '`??`' sign, where each of such errors will be marked by '`??`' sign. Once the location of an error is found in the `.dvi` file, the required correction can be made in the input file.

Warning: `LaTeX Warning: There were multiply-defined labels.`

✓ This is the warning message if the same reference key is defined multiple times through the **\label{}** command, i.e., multiple items are labeled by the same reference key. Unlike in the above two warnings, LaTeX will not show the location of the error in this case. However, checking the list of information generated by the LaTeX compiler, it can be found which reference key is defined multiple times, e.g.,

```
LaTeX Warning: Label 'vcomp' multiply defined.
```

Once the multiply defined reference key is found, the input file can be checked for the locations where it is defined.

23.3 Error Without Any Message

There might be some visual errors, which LaTeX cannot catch and hence the output is generated without any error or warning message. Some of such cases are addressed below:

▷ A table may be printed in a wrong location if wrong syntax is used in **\begin{table}[]** for vertical alignment, say **l**, **c** or **r** in place of **h**, **b**, **t** or **H**.

▷ Strange sectioning numbers will be generated if various sectioning commands are not used in a proper order, e.g., **\subsection{}** without preceded by **\section{}**, or **\subsubsection{}** without preceded by **\subsection{}**.

▷ If a font type command shown in Table 2.1 on page 10 is used for changing fonts of a particular portion, but forgotten to put it in **{}** or to change it by another applicable font type command, the fonts of the entire remaining contents of a document will be changed. For example, **\tt** not in **{}**, or **\scshape** is not followed by **\upshape**.

23.4 Tips for Debugging

It may be difficult to debug a LaTeX input file if it contains a huge number of errors. Sometimes a single error committed in one place may also cause many more errors in the remaining portion of a document, thus making the debugging of the LaTeX input file complicated and cumbersome. Hence, it would be a good practice to proceed as follows:

1. Periodically compile the input file, i.e., without waiting to finish the typing of the entire input file, compile it upon completion of the typing of each segment, say a paragraph or a table.

2. If it becomes difficult to identify or debug an error, first comment the entire doubtful segment with % sign. Then compile the input file repeatedly, each time uncommenting a line or a subsegment in order, until the erroneous line (or subsegment) is detected for debugging.

Hour 24

Exercise

Problem 24.1 Prepare the following table incorporating the \toprule[], \midrule[], \bottomrule[], and \addlinespace[] commands. It shows the Romberg's integration procedure.

Interval (n)	Step size	O(h^2)	O(h^4)	O(h^6)	O(h^8)	O(h^{10})
1	h	$\boldsymbol{I_{1,1}}$				
			$\boldsymbol{I_{1,2}}$			
2	$\frac{h}{2}$	$I_{2,1}$		$\boldsymbol{I_{1,3}}$		
			$I_{2,2}$		$\boldsymbol{I_{1,4}}$	
4	$\frac{h}{4}$	$I_{3,1}$		$I_{2,3}$		$\boldsymbol{I_{1,5}}$
			$I_{3,2}$		$I_{2,4}$	
8	$\frac{h}{8}$	$I_{4,1}$		$I_{3,3}$		
			$I_{4,2}$			
16	$\frac{h}{16}$	$I_{5,1}$				

Problem 24.2 Write the following pseudocode incorporating the \vline command. It is the formation of a random permutation of n.

```
subroutine permutation(n, P)
  for (i = 1 to n) do
  |  Q[i] := i                 // Q is a temporary permutation
  end for

  q := n
  for (i = 1 to n) do          // Loop of the positions
  |  z := rint(1, q)           // z is a random position in (1, q)
  |  P[i] := Q[z]              // Generate an element of the permutation
  |
  |  if (z < q) then
  |  |  for (j = z to q − 1) do
  |  |  |  Q[j] := Q[j + 1]    // Update the temporary permutation
  |  |  end for
  |  |  q := q − 1             // Update the size of the temporary permutation
  |  endif
  end for
end subroutine
```


D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9_24

Problem 24.3 Prepare the following table incorporating **\tabcolsep**, **\rule{0pt}{}**, and **\noindent**, if required. It shows the weekly lecture schedule of an academic Institute.

<table>
<tr><th></th><th>08.00</th><th>09.00</th><th>10.00</th><th>11.00</th><th>12.00</th><th>14.00</th><th>15.00</th><th>16.00</th></tr>
<tr><td>M
O
N</td><td>MTH102NBT
PHY103NAT
PHY102ST
PHY103RT</td><td>MTH102NA
PHY103NB
MTH101S
MTH101R
ESO-II
MTH203R</td><td>D0
ESO-I(T)
CHM201RT</td><td>ESC101NT
ESC102NT
HSS-I</td><td>MTH012NB
PHY103NA
PHY102S
PHY102R
ESO-I
CHM201R</td><td>CHM101LabT
D2/OE</td><td>ESC101N
ESC102N</td><td>MTH102NAT
PHY103NBT
MTH101ST
MTH101RT
ESO-II
MTH203RT</td></tr>
<tr><td>T
U
E</td><td colspan="3">CHM101Lab2
PHY101Lab1
ESC101NLab3
PHY102Lab4
ESO210Lab1</td><td>ESC101N
ESC102N
HSS-1</td><td>D2/OE</td><td colspan="3">ESC101NLab5
ESO214Lab1</td></tr>
<tr><td rowspan="2">W
E
D</td><td colspan="3" rowspan="2">ESC102NLab5
ESO210Lab2</td><td rowspan="2">D0
ESO-I
CHM201R</td><td rowspan="2">MTH102NA
PHY103NB
MTH101S
MTH101R
D2/OE</td><td>D3/OE</td><td>ESO-II
MTH203R</td><td>HSS-I</td></tr>
<tr><td colspan="3">PHY101Lab3
ESC101NLab2
ESC102NLab1</td></tr>
<tr><td>T
H
U</td><td colspan="3">CHM101Lab3
ESC101NLab1
ESC102NLab2
ESO214Lab2</td><td>D0
HSS-1</td><td>MTH102NB
PHY103NA
PHY102S
PHY102R
D3/OE</td><td colspan="3">ESO210Lab3</td></tr>
<tr><td>F
R
I</td><td colspan="3">CHM101Lab1
PHY101Lab2
ESC101NLab4
ESC102NLab3
ESO214Lab2</td><td>MTH102NBT
PHY103NAT
PHY102ST
PHY102RT
D2/OE</td><td>ESC101N
ESC102N
D3/OE</td><td>MTH102NA
PHY103NB
MTH101S
MTH101R
ESO-I
CHM201R</td><td>MTH102NB
PHY103NA
PHY102S
PHY102R
ESO-I(T)
CHM201RT</td><td>MTH102NAT
PHY103NBT
MTH101ST
MTH101RT
ESO-II
MTH203R</td></tr>
</table>

Problem 24.4 Prepare the following table incorporating **\tabcolsep**, **\rule{0pt}{}**, **\noindent**, and **\rowcolor{}** as per requirement. It shows the programme and instructor-wise weekly lecture schedule of another academic Institute.

SN	Subject Code and Title		Room	Lecture Timing	Instructor
		Workshop Practice (CSE)	CW	T3.30, F11.30	Z. Kalita
		Workshop Practice (ECE)	CW	T11.30, Th3.30	S. Kirtania
1	ME103	Workshop Practice (ME)	CW	M11.30, W3.30	P. Kalita
		Workshop Practice (FET+ET)	CW	M3.30, Th11.30	P.P. Dutta1
		Workshop Practice (CE)	CW	W11.30, F3.30	P.P. Dutta2
2	ME201	Solid Mechanics	15/DB	M3.30, T11.30, W11.30(T), Th10.30	Z. Kalita
3	ME202	Fluid Mechanics-I	15/DB	M9.10, T10.30, Th12.30	P. Haloi
4	ME203	Material Science	15/DB	M11.30, W9.10, Th9.10	S. Banerjee
5	ME205	Thermodynamics (ME)	15/DB	M10.30, T9.10, Th10.30, F9.10(T)	P. Kalita
		Thermodynamics (FET)	16/DB	M12.30, T11.30, Th11.30, F2.30(T)	M. Bardalai
6	ME206	ME Laboratory I	MEL	T2.30, Th2.30	S. Kirtania
7	ME301	Dyn. & Vib. of Machinery	11/DB	T2.30, W9.10, Th2.30	S.K. Singh
8	ME302	Measurements & Instrumentation	04/DB	T12.30, W12.30, Th10.30	P. Haloi
9	ME303	Manufacturing Technology II	11/DB	W10.30, Th12.30, F9.10	S. Kashyap
10	ME304	Applied Thermodynamics I	11/DB	T11.30, W11.30(T), F10.30	T.K. Gogoi
11	ME305	Mechanical Design	11/DB	T10.30, W3.30(T), Th9.10,F12.30	P.P. Dutta2
12	ME310	ME Laboratory III	MEL	M2.30, F2.30	SKS, SK2, TKG
13	ME401	Industrial Systems Engineering	04/DB	T9.10, W2.30, F12.30	S. Kashyap
14	ME471	Industrial Summer Training	–	–	–
15	ME534	Mechatronics	04/DB	T12.30, W10.30, F9.10	Z. Kalita
16	ME523	Non-Conventional Energy	04/DB	T10.30,W9.10,F11.30	P.P. Dutta1
17	ME529	Artificial Intelligence in Engg.	04/DB	M2.30, T11.30, F10.30	P.P. Dutta2
18	ME481	Project I	–	–	–
19	ME501	Advanced Solid Mechanics	01/ME	M3.30, W10.30, W2.30(T),Th10.30	S. Kirtania
20	ME504	Failure Analysis of Materials	02/ME	M10.30, Th12.30, F10.30	S. Banerjee
21	ME539	Optimization Techniques in Engg.	02/ME	M12.30, T11.30, F11.30	D. Datta
22	ME541	Advanced Fluid Mechanics	01/ME	M11.30, T10.30, Th2.30, F12.30(T)	T.K. Gogoi
23	ME543	Compressible Flow	01/ME	M2.30(T), W12.30, Th12.30, F10.30	P. Kalita
24	ME561	Exp. Methods for Solids & Fluids	01/ME	T12.30, T3.30(P), W11.30, W3.30(P), Th11.30	PPD1, SKS

Problem 24.5 Prepare the following crossword puzzle table incorporating the **\extrarowheight** and **\cellcolor{}** commands.

1		2		3	■	4		5
	■		■		■		■	
6			■	7				
■	■		■		■	■	■	
8				■	9	10		
	■	■	■	11	■		■	■
12		13			■	14		15
	■		■		■		■	
16			■	17				

Problem 24.6 Write the following conditional expression using the **cases** environment.

$$\bar{\delta} = \begin{cases} \left[2r + (1-2r)\left(\frac{P_i^{\max} - p_{it}}{P_i^{\max} - P_i^{\min}}\right)^{\eta+1}\right]^{\frac{1}{\eta+1}} - 1\,, & \text{if} \quad r < 0.5 \\ 1 - \left[2(1-r) + (2r-1)\left(\frac{p_{it} - P_i^{\min}}{P_i^{\max} - P_i^{\min}}\right)^{\eta+1}\right]^{\frac{1}{\eta+1}} & , \text{otherwise}\,. \end{cases}$$

Problem 24.7 Write the following conditional expression using the **array** environment.

$$\left.\begin{array}{ll} \text{Necessary condition}: \; v_j \in T\,; & \text{if, } X_{ij} = 1\,;\; Y_{il} = 1\,;\; Y_{jl} = 0 \\ & \quad T \subseteq P_m \\ & \quad i, j = 1, 2, \ldots, |V|\,;\; i \neq j \\ & \quad l, m \in \{1, 2, \ldots, K\}\,;\; l \neq m \\ \text{Sufficient condition}: \; \exists r : X_{ir} = 1\,;\; \text{where, } & v_i, v_r \in T \\ & \quad i = 1, 2, \ldots, |V| \\ & \quad r \in \{1, 2, \ldots, |V|\}\,;\; r \neq i \end{array}\right\}$$

Problem 24.8 Write the following expression using the **array** environment.

$$\left[\begin{array}{cccc|cccc} \multicolumn{4}{c|}{\text{Coefficients of}} & \multicolumn{4}{c}{\text{Coefficients of}} \\ \multicolumn{4}{c|}{\underline{\text{Non-basic variables}}} & \multicolumn{4}{c}{\underline{\text{Basic variables}}} \\ a_{11} & a_{12} & \cdots & a_{1,n-m} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2,n-m} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{m,n-m} & 0 & 0 & \cdots & 1 \end{array}\right] = \left\{\begin{array}{c} \text{Resource} \\ \underline{\text{limits}} \\ b_1 \\ b_2 \\ \vdots \\ b_m \end{array}\right\}$$

Problem 24.9 Write the following expression using the **\frac{}{}** command.

$$a = \cfrac{b}{1+\frac{c}{1+\frac{d}{1+\frac{e}{1+\frac{f}{1+h}}}}} + \cfrac{1+\frac{c}{1+\frac{d}{1+\frac{e}{1+\frac{f}{1+h}}}}}{b}$$

Problem 24.10 Prepare the following problem solving procedure incorporating the **tabbing** environment, and nested **flalign** and **array** environments. You can also use **@{}** for adjusting blank space between two columns.

Given: D = 150 mm, F = 7.5 kN, K = 75 N/mm,
G = 81370 N/mm^2, σ_u = 1250 N/mm^2, τ = 30% of σ_u = 375 N/mm^2.

$$\begin{array}{lrl}
\text{Now,} & \text{spring index,} & C = \frac{D}{d} = \frac{150}{d} \Rightarrow d = \frac{150}{C} \\
& \text{shear stress,} & \tau = K_w \frac{8FC}{\pi d^2} = K_w \frac{8FC}{\pi\left(\frac{150}{C}\right)^2} \\
& \Rightarrow & K_w C^3 = \frac{150^2 \pi\tau}{8F} = 441.79 \\
& \Rightarrow & \left(\frac{4C-1}{4C-4} + \frac{0.615}{C}\right) C^3 = 441.79 \\
& \Rightarrow & C \simeq 7.2
\end{array}$$

$$\therefore \quad d = \frac{150}{C} \simeq 21 \text{ mm}$$

$$\text{Also,} \quad K = \frac{Gd^4}{8D^3 N_a}$$

$$\therefore N_a = 7.81 \simeq 8$$

Problem 24.11 Prepare the following page incorporating **\vskip**. Note that the figure is not inserted here as a watermark as discussed in §18.7 on page 177.

'LaTeX Learners Team' is a pseudo-name used in this book for illustrative purpose only. The picture used here as the logo of LaTeX Learners Team is also fake. Existence of the name or the logo somewhere will be just a coincidence. Such a name and a logo were required while citing some examples in this book, such as the watermarking in §18.7 and the slide preparation in §21.3. It is not fare to use some existing name or logo due to copyright issue. So the author felt better to give a pseudo name along with a fake logo. This is to declare again that 'LaTeX Learners Team' is a pseudo-name used in this book for illustrative purpose only. The picture used here as the logo of LaTeX Learners Team is also fake. Existence of the name or the logo somewhere will be just a coincidence. Such a name and a logo were required while citing some examples in this book, such as the watermarking in §18.7 and the slide preparation in §21.3. It is not fare to use some existing name or logo due to copyright issue. So the author felt better to give a pseudo name along with a fake logo.

Problem 24.12 Prepare the following two pages with **twocolumn** and **landscape** options incorporating **\hfill**, **\clearpage**, **\newpage**, etc. The value of **enumi** may also be changed. It is a template for two-sided question paper. Upon printing on both side of a page, it will give a two-sided question paper on a half of the page. If reprinted by turning the page, another copy of the question paper can be obtained on the other half of the page.

TU/ME

TEZPUR UNIVERSITY

Spring Semester End Examination :: 2016
ME 537: COMPUTATIONAL METHODS

Full Marks: 60 **Time: 2 hours**

Answer all questions.

Assume suitable value for any missing data.

---

1. Answer the following questions in short: **[2×6]**
   (a) 1(b)–1(e) below show how the heading of a question paper is to be prepared.
   (b) 1st line: Replace ‘ME’ by the short name of your department in two or three capital letters.
   (c) 3rd line: Replace ‘Spring’ by ‘Autumn’ if it is Autumn semester, and ‘2016’ by applicable year.
   (d) 4th line: Replace ‘ME 537’ by your course code, and ‘COMPUTATIONAL METHODS’ by your course title.
   (e) 7th line: Delete if not required in your course.
   (f) While preparing a paper, your questions are to be typed in 1(a)–1(f).

Please turn over…

---

2. Answer the following questions: **[4×3]**
   (a) Q. No. 1 and Q. No. 2 show how multiple sub-questions of equal marks are to be prepared under a single main question.
   (b) Q. No. 3 below shows how multiple sub-questions of unequal marks are to be prepared under a single main question.
   (c) While preparing a paper, your questions are to be typed in 2(a)–2(c).
3. (a) Q. No. 4 and Q. No. 5 below show how a single question is to be prepared under an individual number. **[1]**
   (b) This is a template for preparing a question paper on the half of an A4-size paper. **[1]**
   (c) If your paper needs both side of a page, as shown in this template, print the two pages either in both-side mode or manually on two sides of a single page. **[3]**
   (d) Repeat the process described in 3(c) by turning the page. Now cut the page into two parts to get two copies of your question paper. **[3]**
   (e) While preparing a paper, your questions are to be typed in 3(a)–3(e). **[4]**
4. Type your question here **[12]**
5. Type your question here **[12]**

~~~~~~~~~~~ ***Best of Luck*** ~~~~~~~~~~~
~~~~~~~~~~~

Problem 24.13 Prepare the following office letter incorporating the **tabularx** environment and **\rule{}{}** command.

To
The Senior Manager
HRD Division, Regional Steel Pvt. Ltd.

Date: May 10, 2016

Subject: **Report of the committee constituted to enquire the genuineness of the grievances lodged by some employees.**
Ref.: Your Office Letter No. RSPL/HRD/2016/Griv/07, dated April 13, 2016.

Dear Sir
With the reference cited above, this is to inform you that after interacting with various personnel and verifying relevant official records, the Committee could complete the said enquiry well within the stipulated time period of 40 days.

The enquiry steps, findings therein, and recommendation of the committee are summarized in the enclosed report for your kind perusal and subsequent action.

Copy to: 1. CMD, RSPL, for information.
2. Enquiry Cell, RSPL, for record.

Encl: As above.

With regards,

(Biplab Rana) Member	(Prabin Singh) Member	(Rakesh Pratap) Coordinator

Appendix A

Symbols and Notations

There are many symbols and notations which may need to be used in a document. Moreover, there exist many special letters used in different languages. All such symbols and letters are to be produced in a LaTeX document through commands. On the other hand, as mentioned in §1.6 on page 7 and given in Table 1.4 on page 8, many keyboard symbols and notations also cannot be used directly in a LaTeX document, but through some commands. Such commonly used symbols, notations, and letters are listed here.

A.1 Text-Mode Accents and Symbols

Text-mode accents and symbols can be produced in running texts without using any special LaTeX package. Commands for producing various non-English accents are given in Table A.1 for the letter 'o'. The commands are equally applicable to other

Table A.1 Text-mode non-English accents

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

letters also, except i and j. In order to remove their dots in accents, i and j are be replaced by \i and \j for producing ı and ȷ), respectively, e.g., \"{\i} for producing ï or \v{\j} for producing ǰ. Note that an accent can be generated without a letter also, e.g., \c{} will produce a (¸) or \v{} will produce a (ˇ). Since \=, \\`, and \' are reserved for special purposes in the tabbing environment, these cannot be used directly for producing accents in that environment (refer §6.2.2 on page 58 for detail).

Table A.2 shows some text-mode non-English symbols, where the commands ?\` and !\`, producing ¿ and ¡, respectively, are noticeable as they are not preceded by any \\. Some other miscellaneous text-mode symbols are given in Table A.3.

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9

Table A.2 Text-mode non-English symbols

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

Table A.3 Text-mode miscellaneous symbols

Symbol	command	Symbol	command	Symbol	command	Symbol	command
#	\#	_	_	†	\dag	£	\pounds
$	\$	ˆ	\^	‡	\ddag	✓	\checkmark
%	\%	&	\&	¶	\P	TEX	\TeX
{	\{	…	\ldots	§	\S	LATEX	\LaTeX
}	\}	⋮	\vdots	©	\copyright		

Many commands of Tables A.1, A.2, and A.3 may not work in math-mode, in which case a command may be applied through **\mbox{}** or **\text{}** (refer §12.1 on page 113 for detail).

A.2 Math-Mode Symbols

The commands required for producing math-mode symbols and letters, like α or β, are defined in separate packages, such as **amssymb**, **amsmath**, and **stmaryrd**. All the three packages can be loaded in the preamble through the single command **\usepackage{amssymb,amsmath,stmaryrd}**. Note that in running texts, a math-mode command is to be inserted in an inline math-mode, such as **$ $**, or **\(\)**, e.g., **β** for producing β or **\(**`x^2+y^2=r^2`**\)** for producing $x^2 + y^2 = r^2$ (refer §11.3 on page 104 for detail). On the other hand, if a symbol is to be produced in boldface, it may be inserted through the **\boldsymbol{}** command, e.g., **$\boldsymbol{\beta}$** in inline, or **\boldsymbol{\beta}** in a math-mode, will produce $\boldsymbol{\beta}$ (β in boldface).

Commands for producing various Greek letters are given in Table A.4 on the next page. The command for a lowercase Greek letter is its name in lowercase preceded by a ****, while that for a uppercase letter is its name with the first letter in uppercase, preceded by a ****.

Various binary operators, relation operators, and arrow symbols are given in Tables A.5, A.6, A.7, A.8, A.9, A.10, A.11, and A.12. Similar to the text-mode accents of Table A.1, some math-mode accents also exist which are given in Table A.13 (an accent command can be used twice to obtain a double accent, e.g., **\acute{\acute{**x**}}** will produce $\acute{\acute{x}}$). Different delimiters, a pair of which is used for enclosing an expression, are given in Table A.14. Moreover, some mathematical functions and other miscellaneous symbols are provided in Tables A.15 and A.16.

Table A.4 Greek letters (math-mode)

Symbol	command	Symbol	command	Symbol	command	Symbol	command
Lowercase		κ	\kappa	υ	\upsilon	Ξ	\Xi
α	\alpha	λ	\lambda	ϕ	\phi	Π	\Pi
β	\beta	μ	\mu	φ	\varphi	Σ	\Sigma
γ	\gamma	ν	\nu	χ	\chi	Υ	\Upsilon
δ	\delta	ξ	\xi	ψ	\psi	Φ	\Phi
ϵ	\epsilon	π	\pi	ω	\omega	Ψ	\Psi
ε	\varepsilon	ϖ	\varpi	**Uppercase**		Ω	\Omega
ζ	\zeta	ρ	\rho	Γ	\Gamma	**AMS Greek**	
η	\eta	ϱ	\varrho	Δ	\Delta	$\digamma$	\digamma
θ	\theta	σ	\sigma	Θ	\Theta	$\varkappa$	\varkappa
ϑ	\vartheta	ς	\varsigma	Λ	\Lambda		
ι	\iota	τ	\tau				

Table A.5 AMS binary operators (math-mode)

Symbol	Command	Symbol	Command	Symbol	Command
$\dotplus$	\dotplus	$\curlyvee$	\curlyvee	$\circledcirc$	\circledcirc
$\ltimes$	\ltimes	$\barwedge$	\barwedge	$\circleddash$	\circleddash
$\rtimes$	\rtimes	$\doublebarwedge$	\doublebarwedge	$\circledast$	\circledast
$\boxdot$	\boxdot	$\curlywedge$	\curlywedge	$\Cap$	\Cap
$\boxplus$	\boxplus	$\smallsetminus$	\smallsetminus	$\Cup$	\Cup
$\boxminus$	\boxminus	$\leftthreetimes$	\leftthreetimes	$\intercal$	\intercal
$\boxtimes$	\boxtimes	$\rightthreetimes$	\rightthreetimes	$\iint\limits_s$	\iint\limits_s
$\veebar$	\veebar	$\divideontimes$	\divideontimes	$\iiint\limits_v$	\iiint\limits_v

Table A.6 Stmaryrd binary operators (math-mode)

Symbol	command	Symbol	command	Symbol	command
$\Yleft$	\Yleft	$\boxast$	\boxast	$\varcurlyvee$	\varcurlyvee
$\Yright$	\Yright	$\leftslice$	\leftslice	$\curlyveeuparrow$	\curlyveeuparrow
$\Yup$	\Yup	$\rightslice$	\rightslice	$\curlyveedownarrow$	\curlyveedownarrow
$\Ydown$	\Ydown	$\oblong$	\oblong	$\varcurlywedge$	\varcurlywedge
$\baro$	\baro	$\obar$	\obar	$\curlywedgeuparrow$	\curlywedgeuparrow
$\fatsemi$	\fatsemi	$\obslash$	\obslash	$\curlywedgedownarrow$	\curlywedgedownarrow
$\merge$	\merge	$\ogreaterthan$	\ogreaterthan	$\binampersand$	\binampersand
$\minuso$	\minuso	$\olessthan$	\olessthan	$\bindnasrepma$	\bindnasrepma
$\moo$	\moo	$\ovee$	\ovee	$\interleave$	\interleave
$\nplus$	\nplus	$\owedge$	\owedge	$\varogreaterthan$	\varogreaterthan
$\bbslash$	\bbslash	$\varbigcirc$	\varbigcirc	$\varolessthan$	\varolessthan
$\sslash$	\sslash	$\varoast$	\varoast	$\vartimes$	\vartimes
$\fatslash$	\fatslash	$\varobar$	\varobar	**Large-sized**	
$\fatbslash$	\fatbslash	$\varobslash$	\varobslash	$\bigbox$	\bigbox
$\talloblong$	\talloblong	$\varoslash$	\varoslash	$\bigcurlyvee$	\bigcurlyvee
$\boxempty$	\boxempty	$\varodot$	\varodot	$\bigcurlywedge$	\bigcurlywedge
$\boxdot$	\boxdot	$\varominus$	\varominus	$\biginterleave$	\biginterleave
$\boxbar$	\boxbar	$\varoplus$	\varoplus	$\bignplus$	\bignplus
$\boxslash$	\boxslash	$\varotimes$	\varotimes	$\bigparallel$	\bigparallel
$\boxbslash$	\boxbslash	$\varovee$	\varovee	$\bigsqcap$	\bigsqcap
$\boxcircle$	\boxcircle	$\varowedge$	\varowedge	$\bigtriangleup$	\bigtriangleup
$\boxbox$	\boxbox	$\varocircle$	\varocircle	$\bigtriangledown$	\bigtriangledown

Table A.7 AMS relation operators (math-mode)

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

Table A.8 Stmaryrd relation operators (math-mode)

Symbol	command	Symbol	command	Symbol	command
$\inplus$	\inplus	$\supsetplus$	\supsetplus	$\ntrianglelefteqslant$	\ntrianglelefteqslant
$\niplus$	\niplus	$\supsetpluseq$	\supsetpluseq	$\ntrianglerighteqslant$	\ntrianglerighteqslant
$\subsetplus$	\subsetplus	$\trianglelefteqslant$	\trianglelefteqslant		
$\subsetpluseq$	\subsetpluseq	$\trianglerighteqslant$	\trianglerighteqslant		

Table A.9 AMS negated relation operators (math-mode)

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

Table A.10 Arrow symbols (math-mode)

Symbol	command	Symbol	command	Symbol	command
$\gets$	\gets	$\leftrightarrows$	\leftrightarrows	$\longleftarrow$	\longleftarrow
$\leftarrow$	\leftarrow	$\leftrightarrow$	\leftrightarrow	$\longrightarrow$	\longrightarrow
$\mapsfrom$	\mapsfrom	$\updownarrow$	\updownarrow	$\longleftrightarrow$	\longleftrightarrow
$\to$	\to	$\Leftrightarrow$	\Leftrightarrow	$\Longleftarrow$	\Longleftarrow
$\rightarrow$	\rightarrow	$\Updownarrow$	\Updownarrow	$\Longrightarrow$	\Longrightarrow
$\mapsto$	\mapsto	$\leadsto$	\leadsto	$\Longleftrightarrow$	\Longleftrightarrow
$\uparrow$	\uparrow	$\nearrow$	\nearrow	$\leftharpoonup$	\leftharpoonup
$\downarrow$	\downarrow	$\swarrow$	\swarrow	$\leftharpoondown$	\leftharpoondown
$\Leftarrow$	\Leftarrow	$\searrow$	\searrow	$\rightharpoonup$	\rightharpoonup
$\Rightarrow$	\Rightarrow	$\nwarrow$	\nwarrow	$\rightharpoondown$	\rightharpoondown
$\Uparrow$	\Uparrow	$\hookleftarrow$	\hookleftarrow	$\longmapsto$	\longmapsto
$\Downarrow$	\Downarrow	$\hookrightarrow$	\hookrightarrow	$\longmapsfrom$	\longmapsfrom

Table A.11 AMS arrow symbols (math-mode)

Symbol	command	Symbol	command
$\dashleftarrow$	\dashleftarrow	$\looparrowleft$	\looparrowleft
$\leftarrowtail$	\leftarrowtail	$\looparrowright$	\looparrowright
$\leftleftarrows$	\leftleftarrows	$\Lsh$	\Lsh
$\twoheadleftarrow$	\twoheadleftarrow	$\Rsh$	\Rsh
$\curvearrowleft$	\curvearrowleft	$\leftrightsquigarrow$	\leftrightsquigarrow
$\circlearrowleft$	\circlearrowleft	$\rightsquigarrow$	\rightsquigarrow
$\dashrightarrow$	\dashrightarrow	$\rightleftarrows$	\rightleftarrows
$\rightarrowtail$	\rightarrowtail	$\Lleftarrow$	\Lleftarrow
$\rightrightarrows$	\rightrightarrows	$\upuparrows$	\upuparrows
$\twoheadrightarrow$	\twoheadrightarrow	$\downdownarrows$	\downdownarrows
$\curvearrowright$	\curvearrowright	$\multimap$	\multimap
$\circlearrowright$	\circlearrowright	$\nleftarrow$	\nleftarrow
$\leftrightharpoons$	\leftrightharpoons	$\nrightarrow$	\nrightarrow
$\rightleftharpoons$	\rightleftharpoons	$\nleftrightarrow$	\nleftrightarrow
$\upharpoonleft$	\upharpoonleft	$\nLeftarrow$	\nLeftarrow
$\upharpoonright$	\upharpoonright	$\nRightarrow$	\nRightarrow
$\downharpoonleft$	\downharpoonleft	$\nLeftrightarrow$	\nLeftrightarrow
$\downharpoonright$	\downharpoonright		

Table A.12 Stmaryrd arrow symbols (math-mode)

Symbol	command	Symbol	command	Symbol	command
$\Mapsfrom$	\Mapsfrom	$\Longmapsto$	\Longmapsto	$\leftarrowtriangle$	\leftarrowtriangle
$\Mapsto$	\Mapsto	$\Longmapsfrom$	\Longmapsfrom	$\rightarrowtriangle$	\rightarrowtriangle
$\nnwarrow$	\nnwarrow	$\longmapsfrom$	\longmapsfrom	$\leftrightarroweq$	\leftrightarroweq
$\nnearrow$	\nnearrow	$\shortuparrow$	\shortuparrow	$\leftrightarrowtriangle$	\leftrightarrowtriangle
$\ssearrow$	\ssearrow	$\shortdownarrow$	\shortdownarrow	$\rrparenthesis$	\rrparenthesis
$\sswarrow$	\sswarrow	$\shortleftarrow$	\shortleftarrow		
$\lightning$	\lightning	$\shortrightarrow$	\shortrightarrow		

Table A.13 Accents and constructs (math-mode)

Symbol	command	Symbol	command	Symbol	command
$\hat{x}$	\hat{x}	$\overline{xyz}$	\overline{xyz}	In accents, replace i and j	
$\check{x}$	\check{x}	$\underline{xyz}$	\underline{xyz}	by \imath and \jmath	
$\breve{x}$	\breve{x}	$\overleftarrow{xyz}$	\overleftarrow{xyz}	$\hat{\imath}$	\hat{\imath}
$\tilde{x}$	\tilde{x}	$\overrightarrow{xyz}$	\overrightarrow{xyz}	$\hat{\jmath}$	\hat{\jmath}
$\acute{x}$	\acute{x}	$\overbrace{xyz}$	\overbrace{xyz}	$\tilde{\imath}$	\tilde{\imath}
$\grave{x}$	\grave{x}	$\underbrace{xyz}$	\underbrace{xyz}	$\tilde{\jmath}$	\tilde{\jmath}
$\dot{x}$	\dot{x}	$\widehat{xyz}$	\widehat{xyz}	$\vec{\imath}$	\vec{\imath}
$\ddot{x}$	\ddot{x}	$\widetilde{xyz}$	\widetilde{xyz}	$\vec{\jmath}$	\vec{\jmath}
$\bar{x}$	\bar{x}	$\sqrt{xyz}$	\sqrt{xyz}	$\acute{\imath}$	\acute{\imath}
$\mathbf{x}$	\vec{x}	$\sqrt[n]{xyz}$	\sqrt[n]{xyz}	$\acute{\jmath}$	\acute{\jmath}
$\mathring{x}$	\mathring{x}	$\frac{xyz}{ab}$	\frac{xyz}{ab}		
x'	x'				

Table A.14 Delimiter symbols (math-mode)

Symbol	command	Symbol	command	Symbol	command	Symbol	command
$[$	[	$\vert$	\vert	$\rmoustache$	\rmoustache	$\Lbag$	\Lbag
$]$	]	$\Vert$	\Vert	AMS delimiters		$\Rbag$	\Rbag
$\lfloor$	\lfloor	Large delimiters		$\llcorner$	\llcorner	$\llfloor$	\llfloor
$\rfloor$	\rfloor	$\arrowvert$	\arrowvert	$\lrcorner$	\lrcorner	$\rrfloor$	\rrfloor
$\lceil$	\lceil	$\Arrowvert$	\Arrowvert	$\ulcorner$	\ulcorner	$\llceil$	\llceil
$\rceil$	\rceil	$\bracevert$	\bracevert	$\urcorner$	\urcorner	$\rrceil$	\rrceil
$\langle$	\langle	$\lgroup$	\lgroup	Stmaryrd delimiters		$\llbracket$	\llbracket
$\rangle$	\rangle	$\rgroup$	\rgroup	$\lbag$	\lbag	$\rrbracket$	\rrbracket
$\backslash$	\backslash	$\lmoustache$	\lmoustache	$\rbag$	\rbag		

Table A.15 Mathematical functions (math-mode)

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

Table A.16 Math-mode miscellaneous symbols

Symbol	command	Symbol	command	Symbol	command	Symbol	command
x'	x'	$\natural$	\natural	AMS **miscellaneous**		$\blacktriangle$	\blacktriangle
∂	\partial	$\prime$	\prime	$\nexists$	\nexists	$\blacktriangledown$	\blacktriangledown
$\aleph$	\aleph	$\emptyset$	\emptyset	$\Bbbk$	\Bbbk	$\vartriangle$	\vartriangle
$\imath$	\imath	∇	\nabla	$\hbar$	\hbar	$\blacksquare$	\blacksquare
$\jmath$	\jmath	$\surd$	\surd	$\hslash$	\hslash	$\lozenge$	\lozenge
ℓ	\ell	$\top$	\top	$\square$	\square	$\blacklozenge$	\blacklozenge
$\wp$	\wp	$\bot$	\bot	$\mho$	\mho	$\varnothing$	\varnothing
$\Re$	\Re	$\Vert$	\\|	$\circledS$	\circledS	$\backprime$	\backprime
$\Im$	\Im	$\sharp$	\sharp	$\Game$	\Game	$\complement$	\complement
∞	\infty	$\clubsuit$	\clubsuit	$\Finv$	\Finv	$\eth$	\eth
$\Box$	\Box	$\diamondsuit$	\diamondsuit	$\bigstar$	\bigstar	AMS **Hebrew**	
$\forall$	\forall	$\heartsuit$	\heartsuit	$\angle$	\angle	$\beth$	\beth
$\exists$	\exists	$\spadesuit$	\spadesuit	$\measuredangle$	\measuredangle	$\daleth$	\daleth
$\neg$	\neg	$\cdots$	\cdots	$\sphericalangle$	\sphericalangle	$\gimel$	\gimel
$\flat$	\flat	$\ddots$	\ddots	$\triangledown$	\triangledown		

Bibliography

1. Goossens M, Mittelbach F, Samarin A (1994) The LaTeX companion. Addison-Wesley Publishing Company, Reading
2. Goossens M, Rahtz S, Mittelbach F (1997) The LaTeX graphics companion. Addison-Wesley Longman Inc., Reading
3. Harvey J, Greenberg A (2000) Simplified introduction to LaTeX. Technical report, Mathematics Department, University of Colorado at Denver. http://www.cudenver.edu/hgreenbe
4. Kopka H, Daly PW (1995) A guide to LaTeX2$_\varepsilon$. Addison-Wesley, Harlow
5. Lamport L (1994) LaTeX: user's guide and reference manual. Pearson Education, Englewood Cliffs
6. Frank M, Michel G, Johannes B, David C, Chris R (2004) The LaTeX companion. Addison-Wesley, Boston
7. Oetiker T, Partl H, Hyna I, Schlegl E (2006) The not so short introduction to LaTeX 2$_\varepsilon$. Technical report. https://www.CTAN.org/tex-archive/info/lshort
8. Tantau T, Wright J, Miletić V (2015) The BEAMER class: user guide for version 3.36. http://bitbucket.org/rivanvx/beamer
9. TeX. http://tex.stackexchange.com

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9

Index

!# . . .

D. Datta, *LaTeX in 24 Hours: A Practical Guide for Scientific Writing*,
DOI 10.1007/978-3-319-47831-9

B b

C c

D d

E e

G g

H h

L l

P p

Q q

R r

S s

T t

U u

V v

W w

X x

Y y

Z z