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 Texts (while **\fbox{**Texts**}** produces 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 **$\cdot$** (·), **$\cdots$** (· · ·), **$\dotsb$** (· · ·), **$\dotsi$** (· ·), **$\dotsm$** (· · ·) and **$\dotsc$** (. . .), 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 22nd April, 2016`' upon loading the **datetime** package without any option as **\usepackage{datetime}** or with the **dayofweek** option as **\usepackage[dayofweek]{datetime}**, or like '`22nd 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 '  ', 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 environment 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 environment 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 ^ 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}**.

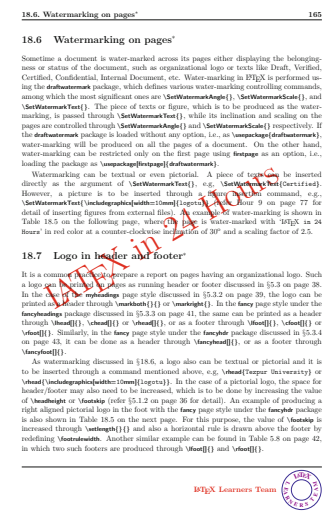
Watermarking can be textual or even pictorial. A piece of texts can be inserted directly as the argument of **\SetWatermarkText{}**, e.g., **\SetWatermarkText{**`Certified`**}**. However, a picture is to be inserted through a figure insertion command, e.g., **\SetWatermarkText{\includegraphics[width=**`10mm`**]{**`logotu`**}}** (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 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.

Table 18.5 Watermarking and logo on pages

LaTeX input

```
\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}
```

Output



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

LaTeX input

```
\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}
```

Output

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

LaTeX input

```
\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}
```

Output

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 ...

Rice	20.00
Oil	60.00
Wheat	25.00

Table 3: Prices.

LaTeX input

```
\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}
```

Output

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 ...

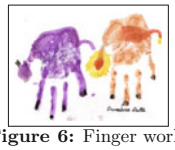


Figure 6: Finger work.

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.