Intermediate $\LaTeX$

October 21, 1997

Mark Senn
mds@ecn.purdue.edu
http://www.ecn.purdue.edu/~mds

Books

The following books were used to help prepare this.

The TEXbook by Donald Knuth. ISBN 0-201-13447-0.

LATEX: A Document Preparation System, 2nd edition, by Leslie Lamport. ISBN 0-201-52983-1.

The LATEX Companion by Michel Goossens, Frank Mittelbach, and Alexander Samarin. ISBN 0-201-54199-8.

A Guide to LATEX 2_ε, 2nd edition, by Helmut Kopka and Patrick W. Daly. ISBN 0-201-42777-X.

The Chicago Manual of Style. a standard reference tool for authors, editors, copywriters, and proofreaders.

A Complete Example

```
\documentclass{article}
\begin{document}
\title{Short Example}
\author{Mark Senn}
\maketitle

This is
the first paragraph.
Note
the indentation.

This is
the second paragraph.
\end{document}
```

Short Example

Mark Senn

October 21, 1997

This is the first paragraph. Note the indentation.

This is the second paragraph.

1

Unix Tip: edit, latex, preview

To switch between editing, latexing, and previewing easily:

1. Make or choose an window in which you'll edit files. Make or choose another window in which you'll run LaTeX.

2. Get in *filename*`.tex` in your favorite text editor in the edit window.

3. Make desired changes in edit window and write file (type '`:w`' in vi).

4. Type '`latex` *filename*' in LaTeX window. If you've LaTeX'ed the file recently you may be able to run LaTeX again by typing '`!la`'.

5. If this is your first time at this step type '`xdvi` *filename* &' in the LaTeX window. Resize and position this window as wanted.

If not first time at this step just move mouse in the xdvi window. The xdvi window will be updated with the new version of the document.

6. If the document needs more work go to Step 3.

Document Parts

Some commands can only be used in a 'preamble' or 'body' of a document. Here is where the preamble and body are:

```
\documentclass
preamble
\begin{document}
body
\end{document}
```


Portrait/Landscape

Normally LaTeX prints across the narrow part of the page. To have LaTeX print across the wide part of the page tell LaTeX about the new width and height and tell the printer to print it sideways.

(Don't use a dvipr program option to tell the printer to print sideways. Sooner or later you'll forget to use the option. Using the '`\special{papersize...}`' will print sideways on the page automatically.)

```
  % tell LaTeX height of text
\setlength{\textheight}{6truein}
  % tell LaTeX width of text
\setlength{\textwidth}{9truein}
  % tell printer to print it sideways
\special{papersize=11in,8.5in}
```

Reading Other Files **also see "Document Parts" (slide 7)**

`\input`{*filename*}

> The `\input` statement causes *filename* to be read immediately. The `\input` statement can occur anywhere in the file—even before the `\documentclass` command. It does not start a new page.

`\include`{*filename*}

> The `\include` statement always starts a new page. It can only occur in the body of a document.

`\includeonly`{*filename1*, . . .}

> The '`\includeonly`' statement can only occur in the preamble of a document. Only the filename(s) listed in the `\includeonly` statement are processed during this LaTeX run.

Aligned equations

The `eqnarray` and `eqnarray*` environments are used to align equations (make equal signs line up). See a LaTeX book for more information.

The `amstex` package defines the following environments for creating aligned equations: `align`, `alignat`, `xalignat`, `equation`. See *The LaTeX Companion* for more details.

Encapsulated PostScript Files

Many programs can produce an Encapsulated PostScript File. Encapsulated Postscript is a language used for graphics (graphs, charts, pictures, etc.)

To get ready to include EPSF graphics in a LaTeX document use

```
\documentclass{class}
\usepackage{epsf}
```

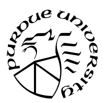
To insert a graphic use the `\epsffile` command. Normally the graphic comes out in its natural size. You can set `\epsfxsize` *or* `\epsfysize` to force a particular width or height to scale the graphic proportionally. Setting `\epsfxsize` *and* `\epsfysize` forces the graphic to be that width and height.

Encapsulated PostScript Files (part 2)

I typed

```
\setlength{\epsfxsize}{0.4in}
\epsffile{griffin.eps}
```

to get



and

```
\setlength{\epsfxsize}{1in}\setlength{\epsfysize}{0.4in}
\epsffile{griffin.eps}
```

to get

Floats

Figures and tables "float" to their positions in the output. Normally LaTeX tries to put the figure at the top of a page, at the bottom of a page, or on a page containing just floats.

The general form of a float command is

> `\begin{`*float*`}[`*options*`]`

where *float* is `figure` or `table` and placement *options* include:

> `b` (bottom)
> `h` (here),
> `p` (page with only floats on it),
> `t` (top of page),
> `!` (suspend rules to emphasize I really want it here)

Figure **also see "Encapsulated PostScript Files" (slide 12)**

The `figure` floating environment is used for figures.

I typed

```
\begin{figure}
  \caption{Test Encapsulated PostScript File}
  \setlength{\epsfxsize}{0.5in}
  \epsffile{griffin.eps}
\end{figure}
```

to get

Figure 1: Test Encapsulated PostScript File

Table

The `table` floating environment is used for tables. I typed

```
\begin{table}
  \caption{Test Table}
  \begin{tabular}{l|c|r}
    l& c& r\\
    left& center& right\\
  \end{tabular}
\end{table}
```

to get

Table 1: Test Table

l	c	r
left	center	right

List Making Environments

LaTeX provides three types of list making environments: `itemize` for bulleted (•) lists, `enumerate` for numbered (1,2,...) lists, and `description` for labeled lists.

Each type of list starts with `\begin`{*environment*} and ends with `\end`{*environment*}. New items in `itemize` and `enumerate` lists begin with `\item`. New items in `description` lists begin with `\item[`*label*`]`. Lists can be nested. See the example on the next slide.

In addition you can define your own kind of list using the `list` environment. This can be pretty tricky though. See a LaTeX book for more information.

List Making Environments (part 2)

I typed

```
\begin{description}
  \item[breakfast] The morning meal.
  \begin{enumerate}
    \item Wakes me up
    \item Gives me strength.
  end{enumerate}
  \item[lunch] The midday meal.
end{description}
```

to get

breakfast The morning meal.

1. Wakes me up.
2. Gives me strength.

lunch The midday meal.

Verbatim **also see "Verb" (slide 16)**

To print *whole lines* of text verbatim, instead of having it interpreted by LaTeX, use the verbatim environment. The verbatim environment prints its output in a 'typewriter' font.

I typed

```
\begin{verbatim}
This is some verbatim input.
\setlength{\textheight}{6truein}
\end{verbatim}
```

to get

```
This is some verbatim input.
\setlength{\textheight}{6truein}
```

'`\begin{verbatim*}`' prints '␣'. '`\begin{verbatim}`' prints ' '.

Verb   **also see "Verbatim" (slide 17)**

To print *partial lines* of text verbatim, instead of having it interpreted by LaTeX, use the '`\verb`' command.

Surround the text you want printed verbatim with a character that dosn't occur in the text. (My first choice is '`+`'. If the text uses '`+`', my second choice is '`-`'.)

'`\verb*`' prints '␣'. '`\verb`' prints ' '.

EXAMPLE: '`\verb+nothing special+`' gives '`nothing special`'.

EXAMPLE: '`\verb+\hello there+`' gives '`\hello there`'.

EXAMPLE: '`\verb*-$a = b + c$-`' gives '`$a␣=␣b␣+␣c$`'.

Special Symbols

The AMS (American Mathematical Society) fonts have been installed.

To use the most common symbols in LaTeX 2_ε put a `\usepackage{amssymb}` statement right after your `\documentclass` command.

This allows you to use hundreds of math special symbols including $\therefore$ (`\therefore`), $\mathbb{R}$ (`\mathbb{R}`), etc.

I'm working on complete instructions for this and will send you email when done if you're on the signup sheet. Otherwise send me email at mds@ecn.purdue.edu and I'll contact you when they're done.

Defining Commands

Using LaTeX one can define (using `\newcommand`) or redefine (using `\renewcommand`) commands.

By setting up a few of your own commands you can sometimes greatly simplify what you need to type.

For example, typing '`\newcommand{\xn}{$x_n$}`' makes '`\xn`' a shorthand for '`$x_n$`' (x_n).

This definition can be made more flexible by typing '`\renewcommand{\xn}{\ensuremath{x_n}}`' so '`\xn`' is a shorthand for '`\ensuremath{x_n}`' (x_n). The '`\ensuremath`' lets the command be used in math or text mode—it goes into and out of math mode if not in math mode and does nothing if already in math mode. Since '`\xn`' was already defined, '`\renewcommand`' was used to redefine it.

Defining Commands (part 2)

If you would just like to make a shorthand for a long built-in command name use the '`\let`' command. For example, typing '`\let\en=\ensuremath`' makes '`\en`' be a shorthand for '`\ensuremath`'.

Using the above '`\en`' definition, typing '`\renewcommand{\xn}{\en{x_1},\␣\en{x_2},\␣\ldots,~\en{x_n}}`' makes '`\xn`' a shorthand for '`\en{x_1},~\en{x_2},\␣\ldots,~\en{x_n}`' (x_1, x_2, …, x_n).

In this example the '`\␣`' is used to get the correct spacing (remember, ordinary spaces are ignored in math mode) and the '`~`' is used to tie the '…,' to the 'x_n' so they won't be broken between lines.

Defining Commands (part 3)

We know how to have make an easy-to-type command name be a shorthand for input that's more difficult to type in. The next logical step is to be able to create commands with substitutable parameters.

Typing '`\renewcommand{\xn}[1]{\en{#1_n}}`' makes '`\xn ①`' be a shorthand for '`\en{①_n}`' (①_n). The '`[1]`' means that there is one parameter. The '`#1`' is substituted by the first parameter when LaTeX is run.

Example: '`\xn a`' gives 'a_n'.

Typing '`\renewcommand{\xn}[2]{\en{#1_#2}}` makes '`\xn ①②`' be a shorthand for '`\en{①_②}`' ($\text{①}_{\text{②}}$). The '`[2]`' means that there are two parameters. The '`#1`' is substituted by the first parameter and '`#2`' is substituted by the second parameter when LaTeX is run.

Example: '`\xn ab`' gives 'a_b'.

Defining Commands (part 4)

Harry Shearer has decided to reprise his role of Eddie Haskell in a new "Leave it to Beaver" show. We define

```
\newcommand{com}[1]{Good morning Mrs.~Cleaver.
  My, but your #1 today.}
```

so we can type, for example, `\compliment{hair looks lovely}` to compliment Mrs. Cleaver on her hair.

Later, Harry quits the show and Sylvester Stallone gets the role. We can keep all uses of `\com` in the script the same and only change the definition of `\com` to

```
\newcommand{com}[1]{Yo, Cleaver, your #1.}
```

Defining commands for math or text patterns that happen frequently in the input can sometimes save a *lot* of work.

BibTeX

The BibTeX program can be used to help prepare bibliographies. In short, here is how it works.

You prepare a '`.bib`' file that contains a list of works you may reference. Each entry contains a key, author, title, publisher, etc. information.

In your document, when you want to cite a reference just type '`\cite`{*key*}' where *key* is the key for that entry in the `.bib` file you want to cite. BibTeX will insert the appropriate citation to put in the text (e.g. [10] for the tenth entry in the bibliography) and will put the information in the bibliography automatically in the correct format for this document. Dozens of different bibliography styles are supported.

If you use references check this out!