A Short Introduction to \LaTeX

David J. Eck

October 22, 2003

Abstract

This paper is a very short introduction to the essentials of \LaTeX, a document-preparation system that is an alternative to typical word-processing programs. \LaTeX is particularly good at mathematical typesetting, but it has features that make it nice for almost any kind of writing. Most of the information in this paper is taken from A Guide to \LaTeX, by Helmut Kopka & Patrick W. Daly.

1 The Basics of \LaTeX

The \TeX typesetting system was created by the well-known computer scientist Donald Knuth because he wasn’t satisfied with the way his books looked when they were published by existing means. \TeX produces beautiful output, especially of mathematics, but it is hard to use directly. Mathematician Leslie Lamport built \LaTeX on top of \TeX to provide easier access to fine typesetting.

Writing papers with \LaTeX is very different from writing with a word processor. With a word processor, you have to be concerned with many minute details of the format and layout of your paper. \LaTeX takes care of many of these details for you. On the other hand, to use \LaTeX, you have to learn something that will look to many people like a programming language (although HTML, the layout language of the Web, is a better analogy). Fortunately, you can learn the basics in a very short time. This short paper aims to cover enough to get you to the point of producing attractive documents with \LaTeX.

Maybe the first thing to note is that the actual typing of a \LaTeX paper is done in a plain text editor rather than in a word-processing program. The text file is then processed to produce the actual output. The output can then be printed or can be viewed on-screen with a special viewing program. You can also convert \LaTeX output to other formats such as HTML or PDF.
1.1 Special Characters

Certain characters have special meaning in \LaTeX. Typing these characters doesn’t make them appear in the output. For example, the character % is used to add comments to \LaTeX input files: A % and everything that follows it on the line is ignored by \LaTeX.

The most important special character is the backslash, \. The backslash is used to give special commands to \LaTeX. For example, the \LaTeX logo itself is actually produced by the command \LaTeX. A command consists of a backslash followed by a sequence of letters. Also, a command can consist of a backslash followed by a single non-letter character. For example, if you want the special character % to appear in the output, you can type it as %\%. % is a command that tells \LaTeX to put a % into the output.

Other special characters include $, #, &, \_, \{, and \}. The dollar sign is particularly important in \LaTeX: It is used to enclose mathematics. For example, the mathematical inequality $x < y$ is typed as $x<y$. This is called math mode. There are three special characters that only come out correctly in math mode: <, >, and |. These characters can be typed between dollar signs. For example: $<$. The backslash itself can be generated using the command \backslash in math mode; that is, $\backslash$ yields \ in the output. There are two more special characters, ~ and ^, that we will get to later.1

Some characters that appear in documents can’t be typed at all. An example is the left double quote mark, "\. In \LaTeX, this symbol is inserted into the output by typing two single left quotes: ‘‘. You can type a right quote as two right single quotes, ’’ (although in fact typing the usual double quote character will give the same thing). Similarly, there are actually three types of dashes: the hyphen (-), the n-dash (—) that is used in number ranges such as 2003—2004, and the m-dash (—) that is used to separate text—like this—from the rest of the sentence. In \LaTeX, you type -, —, and — as -, —, and ——.

There are many more special characters, especially when you consider mathematical symbols. These are entered into \LaTeX output using special commands. For example, the command \copyright gives ©, and \dag gives †. There are a lot of commands that are valid only in math mode, and we will see some of these later. The command \dots gives the three dots that are usually used to represent an ellipsis; you need this because typing three periods gives ..., and that doesn’t look as good as . . .

And finally, there are characters that are accented or otherwise decorated, such as the ô in coöperate. The most common accents are input as

---

1All this might seem like a big deal, but in fact, these characters are rarely used in normal text. You should just be aware that some characters have special meaning so you aren’t surprised when you do happen to use them.
single-character commands in front of the letter that is to be accented. Type \'e, \'e, \"u, \"n, \"o, and \=a, for é, è, ü, ū, ô, and ä.

1.2 Commands and Environments

As I’ve already mentioned, \LaTeX commands begin with the special character \ . Some commands can be followed by parameters that provide information used by the command. An example is \footnote, which is used to insert a footnote into a paper. The text of the footnote is the parameter to the command. The parameter follows the command and is enclosed between { and }. In fact, you can leave out the { and }, but in that case the parameter will be the single character that follows the command. This is what is done with the accenting commands above, so that \'e is equivalent to \'{e}.

Some commands can include extra, optional information that is enclosed between [ and ] and placed between the command and the parameter. For example: \documentclass[11pt]{article}. And it’s possible for a command to have several parameters, each enclosed in its own pair of braces. For example: \setlength{\parindent}{2cm}.

There is one annoying thing about commands that consist of a sequence of letters: \LaTeX will ignore any spaces that follow the command. This means that if you type “\LaTeX is fun!”, it will come out on the page as “\LaTeXis fun!”}. To fix this, you can add an explicit space command. The command “\ ” (that’s a backslash followed by a space) represents a space that \LaTeX won’t ignore. So, to get “\LaTeX is fun!” to come out right, you should type “\LaTeX \ is fun!”.

\LaTeX gets a lot of mileage out of two general commands, \begin{name} and \end{name}, where name is the name of an “environment” such as itemize or quotation. These commands always occur in a begin/end pair, using the same name in both places. Anything between begin and end is treated in a special way that depends on which particular environment you are using. For example, \begin{center} ... \end{center} is used to center things horizontally on the page. Many environments have special commands that only apply inside that environment. We will see some examples of environments later.

1.3 Spacing

In \LaTeX input, words are separated by spaces, and paragraphs are separated by blank lines. However, adding extra spaces will not increase the space.

---

\footnote{When you insert a footnote, a little superscript appears in the text, and the footnote is placed at the bottom of the page. This footnote was created by typing: \footnote{When you insert a footnote...}}
between words in the output, and adding extra blank lines will not increase the space between paragraphs. Spaces at the beginning of a paragraph are ignored—paragraph indentation is inserted automatically whether there are spaces in the input or not.

There are commands that you can use to insert extra space in the output. To insert some extra vertical space, for example between paragraphs, use the commands \smallskip, \medskip, and \bigskip. There is a \medskip just before this paragraph, so you can get an idea of the size. You can insert a specific amount of vertical space with a command such as \vskip{0.5 in}. The units of measurement—which can be in for inches, cm for centimeters, or pt for points—must be specified.

If you would like to leave out the indentation at the beginning of a particular paragraph, put the command \noindent at the beginning of the paragraph. You can insert horizontal space with \hspace, which works similarly to \vspace. The commands \quad and \qquad also insert horizontal space. (“Quad” is a printer’s term; \quad is a double quad.)

\LaTeX also ignores line breaks inside paragraphs. The text in the paragraph is automatically broken into lines in an optimal way. \LaTeX even breaks words and inserts hyphens if necessary. However, you can force a line break by typing the command \\. This command works almost anywhere. For example, you can use it in the title of the document, even though that is not an ordinary paragraph.

Sometimes, you want to tell \LaTeX not to break a line at a certain space. For this, you can use the ~ character as a non-breaking space. The ~ does not appear in the output—it becomes a space. However, \LaTeX will definitely avoid breaking the line at that space.

1.4 Text Styles

Italic, bold-face, and other text styles are easy in \LaTeX. For example, you can use \textit{this text in italic} to output this text in italic. Similarly, the \textbf will apply boldface to its parameter. Here is a table of text styles that can be used:

<table>
<thead>
<tr>
<th>Command</th>
<th>Meaning</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{sample text}</td>
<td>Italics</td>
<td>sample text</td>
</tr>
<tr>
<td>\textbf{sample text}</td>
<td>Bold face</td>
<td>sample text</td>
</tr>
<tr>
<td>\textsl{sample text}</td>
<td>Slanted</td>
<td>sample text</td>
</tr>
<tr>
<td>\texttt{sample text}</td>
<td>Typewriter style</td>
<td>sample text</td>
</tr>
</tbody>
</table>

You can combine these commands to get effects such as boldface that is also italic or slanted.
1.5 Verbatim Output

Sometimes you would like \LaTeX to output things exactly as you type them, without interpreting special characters in the input. This would be particularly useful when including some programming source code in your document, for example. You can do this with the \texttt{verbatim} environment. That is, if you put the command \texttt{\begin{verbatim}} at the beginning of some text and \texttt{\end{verbatim}} at the end, then \LaTeX will reproduce the text in the output, in typewriter style, exactly as it is typed in the input. Spaces and ends-of-line are copied to the output as well as other special characters.

If you want to include just a few words of verbatim output in the middle of a paragraph (as I do many times in this paper), you can use \texttt{\verb}. This command takes a special form: \texttt{\verb=verbatim text=}. The text between the two equal signs is copied directly to the output in typewriter style. This text can contain any character except an equal sign, but cannot extend over more than one line. (If the text you want to output does contain a =, you can replace the = in the \texttt{\verb} command with any other character.) One problem with \texttt{\verb} is that it doesn’t work inside the parameters of other commands, such as \texttt{\footnote}.

2 Mathematics

One of the main advantages of using \LaTeX is that it so good at typesetting mathematics. Mathematical input in \LaTeX is typed in what is called \textit{math mode}. To type math that is to appear as part of a paragraph or other piece of text, enclose the math input between a pair of dollar signs, such as $x < y$. If you use a pair of double dollar signs, $$x < y$$, you get \textit{display math mode} where the mathematics is displayed on a line by itself and centered horizontally on the page. A careful mathematical typist will use math mode even for a thing as simple as a one-character variable. This is because a variable such as $x$ should properly be printed in italics. Math mode will take care of this. This means that you should type the variable as $x$ rather than simply as $x$.

2.1 Basic Math

In general, quality typeset mathematics looks quite a bit different than plain typeset text. \LaTeX will take care of most of the little details—such as the amount of space to leave around a plus sign—for you. Since \LaTeX decides on the proper spacing, it \textit{completely ignores spaces} in math mode. (If you are in math mode and want to include some regular text with regular spacing, you can use the \texttt{\text} command. The parameter of this command is typeset as regular text.) You can also use the “\ ” command (backslash followed by a
space) in math mode for a space that won’t be ignored. And there are a few commands for adding smaller amounts of space in math mode: \, and \;.

Mathematical operators such as +, −, *, /, <, =, and > will behave as you expect in math mode. Set braces in math mode, as in regular text mode, must be types as \{ and \}. The Greek letters are specified by commands that are the same as the name of the letter: \alpha, \beta, \gamma, \delta, \epsilon, \theta, and γ, for example. Capitalized Greek letters use capitalized commands so that Σ and Φ are typed as \Sigma and \Phi. There are many commands that produce special mathematical characters. A sample is shown in this table:

<table>
<thead>
<tr>
<th>\le</th>
<th>≤</th>
<th>\ge</th>
<th>≥</th>
<th>\ne</th>
<th>≠</th>
</tr>
</thead>
<tbody>
<tr>
<td>\to</td>
<td>→</td>
<td>\cup</td>
<td>∪</td>
<td>\cap</td>
<td>∩</td>
</tr>
<tr>
<td>\in</td>
<td>∈</td>
<td>\subset</td>
<td>⊂</td>
<td>\emptyset</td>
<td>∅</td>
</tr>
<tr>
<td>\wedge</td>
<td>∧</td>
<td>\vee</td>
<td>∨</td>
<td>\lnot</td>
<td>¬</td>
</tr>
<tr>
<td>\forall</td>
<td>∀</td>
<td>\exists</td>
<td>∃</td>
<td>\oplus</td>
<td>⊕</td>
</tr>
<tr>
<td>\rightarrow</td>
<td>→</td>
<td>\Rightarrow</td>
<td>⇒</td>
<td>\therefore</td>
<td>∴</td>
</tr>
<tr>
<td>\infty</td>
<td>∞</td>
<td>\partial</td>
<td>∂</td>
<td>\approx</td>
<td>≈</td>
</tr>
</tbody>
</table>

Subscripts and superscripts are specified using the special characters _ and ^. These commands act like commands whose parameters are put into the subscript and superscript positions. For example, $a_{i}$ is typeset as $a_{i}$. You can combine subscripts and superscripts: $x_{i}^{n}$ comes out as $x_{i}^{n}$. (Note that you can omit the { and } around the parameter if the parameter consists of a single character. However, be aware that if you type $x^{17}$, you will get $x^{1}$ because only the 1 is in the superscript.) Subscripts and superscripts can be complicated and can even include other levels of scripting. So, you can have $x_{b^{2}+a_{i}^{j}}$, which is typed as $x_{b^{2}+a_{i}^{j}}$.

To take the square root of something, use \sqrt, which applies the square root sign to its parameter. For example, $\sqrt{x^{17}}$ yields $x^{17}$. You can combine subscripts and superscripts: $x_{i}^{n}$ comes out as $x_{i}^{n}$. (Note that you can omit the { and } around the parameter if the parameter consists of a single character. However, be aware that if you type $x^{17}$, you will get $x^{17}$ because only the 1 is in the superscript.) Subscripts and superscripts can be complicated and can even include other levels of scripting. So, you can have $x_{b^{2}+a_{i}^{j}}$, which is typed as $x_{b^{2}+a_{i}^{j}}$.

To take the square root of something, use \sqrt, which applies the square root sign to its parameter. For example, type $\sqrt{x^{2}+1}$ to get $\sqrt{x^{2}+1}$. The \frac command takes two parameters and makes them into a fraction in which the first parameter is the numerator and the second is the denominator. So, $\frac{n+1}{n}$ will give \frac{n+1}{n}. \frac is the first example we’ve encountered that looks different in display math mode. Typing \frac{n+1}{n} will give the displayed fraction

\[
\frac{n+1}{n}
\]

in which the characters are much larger. In some cases, you would like to have some in-line math look like displayed math. You can get this effect
by adding the \texttt{\textbackslash displaystyle} command. So, you can get \( \frac{n+1}{n} \) by typing 
\texttt{$\frac{n+1}{n}$}. Similarly, you can use \texttt{\textbackslash textstyle} to force display mode math to be rendered in the style of in-line math.

The summation, product, and integral signs are written as \texttt{\textbackslash sum}, \texttt{\textbackslash prod}, and \texttt{\textbackslash int}. Subscripts and superscripts work in a special way with these symbols to add the usual “decorations” at the top and bottom of the symbol. For example, \texttt{$\sum_{n=1}^{\infty} \frac{x^n}{n!} = e^x$} is rendered as

\[
\sum_{n=1}^{\infty} \frac{x^n}{n!} = e^x
\]

Rendered as in-line math, this would be \( \sum_{n=1}^{\infty} \frac{x^n}{n!} = e^x \). This is one case where I definitely prefer to use display style.

The names of mathematical functions such as \texttt{\textbackslash sin} and \texttt{\textbackslash ln} should properly not be in italics. \LaTeX defines commands for many function names so that they can be typeset properly. So, for example, you should type \texttt{\textbackslash sin(x)} to get \( \sin(x) \). Certain commands, such as \texttt{\textbackslash lim}, use subscripts in the same way as the \texttt{\textbackslash sum} command. For example, \texttt{$\lim_{x \to \infty} x^{-x} = 0$} gives

\[
\lim_{x \to \infty} x^{-x} = 0
\]

Mathematics makes extensive use of bracketing symbols such as (, ), [ ], { }, |, and \|. (The \{, \}, and \| are typed as \texttt{\{, \} and \|. You can only use | and \| in math mode.) All these symbols should change size according to the size of the stuff inside the brackets. \LaTeX uses the commands \texttt{\left} and \texttt{\right} to make bracketing symbols that will change size. These commands are used with bracketing symbols: \texttt{\left(, \right), \left\{, \right\}, \left\|, \right\|}. \texttt{\left} and \texttt{\right} must always occur as a pair, although the type of bracketing symbol on the left and right don’t have to match. \LaTeX will choose the appropriate size of the brackets.\footnote{If you are not happy with the size, you can use commands like \texttt{\big(}, \texttt{\Big(}, \texttt{\bigg(}, and \texttt{\Bigg(} to get various specific sizes.}

There are many other, more sophisticated aspects of mathematical typing. Some of these are provided by \LaTeX environments. I give one example here. If you want to learn more, you will have to consult a reference. The array environment can be used to typeset arrays. An array environment begins with something like \texttt{\begin{array}\{ccc\}}. The “ccc” says that the array will have three columns and that the items of the array will be centered in each column. Replace a c with an l or an r to get left- or right-justified items. The end of the array is marked with \texttt{\end{array}). In the array environment, lines of the array are separated by \texttt{\\\}, and the items on a line are separated by \&. The array is not automatically enclosed in parentheses. Use \texttt{\left(} and \texttt{\right)} to get that effect. Here is an array example:

\[
\begin{array}{ccc}
\end{array}
\]
This array uses \texttt{\cdots} and \texttt{\vdots} to represent groups of three dots. These commands are treated just like other array items. The blank space in the middle of the array is an empty item. Note that in the source code this blank item is indicated by two \&'s with nothing between them.

### 3 Document Structure

We’ve covered some of the essential facts about what you can put into a \LaTeX{} document, but we still need to look at how the document must be organized as a whole.

Every \LaTeX{} document should begin (except possibly for some comments) with a \texttt{\documentclass} command. This command has a parameter that specifies the type of document: article, report, letter, or book. The command can also specify some options for the document. The basic font size can be specified with the 10pt, 11pt, or 12pt options. The default is 10pt. Specify 11pt or 12pt for larger-sized characters. The \texttt{twocolumn} option will cause text to be set in a double-column format. Here is a typical example:

\begin{verbatim}
\documentclass[11pt,twocolumn]{article}
\end{verbatim}

Following the \texttt{\documentclass} command comes the \texttt{preamble}, which can contain commands that influence the contents or appearance of the document. The preamble does not itself produce any output. The preamble often starts with some \texttt{\usepackage} commands. A package makes available a set of extra, non-standard commands for use in the document. For example, \texttt{\usepackage{amsfonts}} and \texttt{\usepackage{amssymb}} make a large number of extra mathematical symbols available. (In fact, the \texttt{\therefore} command, which was listed in a table earlier in this paper, requires these packages.)

The preamble command \texttt{\usepackage[dvips]{graphics}} makes it possible to include graphics, as discussed later in this paper.

Commands that specify the title, author, and date of the paper are often given in the preamble. For example:
The information in these commands is actually used later, in the body of the paper, when you give the \texttt{\maketitle} command.

Page layout in a document is partially controlled by the value assigned to various lengths, such as the size of the margins and the distance from one line of text to the next. The preamble can contain commands for customizing some of these lengths using the \texttt{\setlength} command. For example, for this paper, I use the commands

\begin{verbatim}
\setlength{\topmargin}{0in}
\setlength{\textheight}{8in}
\end{verbatim}


to increase the vertical size of the text on the page from its default value. The first parameter is the name of the length that is being set. The second parameter is the value, which must be given in units such as in, cm, or pt. The distance between lines of text—the so-called \textit{baselineskip}—is handled a little differently. You have to specify a factor by which the default baselineskip is multiplied. A factor of 2 will give double-spaced text, although a factor of about 1.8 probably looks better. The command that you would use in the preamble for double-spacing is: \texttt{\renewcommand{\baselinestretch}{1.8}}. You can actually use this command anywhere in your document to change the baselineskip for the following paragraphs.

\LaTeX is actually a programming language, in which you can define your own commands. The preamble often includes definitions of new commands that will be used in the paper. However, this aspect of \LaTeX will not be discussed here.

After the preamble comes the document content. The body of the document begins with \texttt{\begin{document}} and ends with \texttt{\end{document}}. The actual content of the document is typed between these commands. A document of any length is typically divided into parts, chapters, sections, subsections, and subsubsections. In \LaTeX, an article can have only sections, subsections, and subsubsections, while books and reports can also have parts and chapters. To start a new section, just say \texttt{\section{title}} where \textit{title} is the title of the section. The same thing works for parts, chapters, subsections, and subsubsections. You can use \texttt{\textbackslash \textbackslash} in a title to force a line break.

Sections, subsections, and so on will be automatically numbered for you by \LaTeX. If you don’t want the numbering, you can add a * after the name of the command: \texttt{\section*{title}}. This will suppress the numbering.
If you include the command \pagestyle{headings} in the preamble of the document, \LaTeX{} will add running headings to the top of each page. The headings will contain the section or chapter titles. Sometimes, the heading will be too long to fit. In that case, you can provide a short title for the headings. The short title is given as an option to the command: \section[short-title]{long-title}.

4 Fancy Stuff

\LaTeX{} is used to produce typeset documents of great complexity, and this paper can only give you an idea of the possibilities. In this last section, I will look briefly at a few of \LaTeX{}’s other features.

4.1 Lists

Numbered and bulletted lists are very common in documents. The \texttt{enumerate} and \texttt{itemize} environments are used for creating such lists. Thus, a numbered list begins with \begin{enumerate} and ends with \end{enumerate}. The items of the list go between these. The start of each item should be marked with the command \texttt{item}, which can be thought of as generating the item number. \begin{itemize} and \end{itemize} are used in exactly the same way to create bulletted lists. Lists can be nested inside one another, and \LaTeX{} will indent the lists properly to show the nesting. For numbered lists, the style of numbering (numbers, letters, or roman numerals) changes depending on the level of nesting.

4.2 Tables

Tables are very similar to arrays in mathematics, but tables often contain words and have lines separating rows and columns. \LaTeX{} has a \texttt{tabular} environment that is almost identical to the array environment. As with arrays, the command that begins the environment has an extra parameter that specifies the format of the columns. This format uses \texttt{l}, \texttt{r}, and \texttt{c} to for left-, right- and center-justified columns. It can also include \texttt{|} characters to indicate where vertical lines are to be drawn in the table. For example, \begin{tabular}{|r||cc|} \hline \texttt{\begin{tabular}{|r||cc|}} \end{tabular}} \end{tabular} Will produce a table with three columns. The entries in the first column are right-justified; the other two columns are center-justified. There is a vertical line on each side of the table, a double vertical line between the first and second column, and no line between the second column and the third.

In the data for the table, the end of a row is marked with \texttt{\}. Items on a row are separated with \texttt{&}. You can also put the command \texttt{\hline} before the first row, after the last row, or between rows to indicate places where
a horizontal line is supposed to be drawn across the table. If you want to
draw a horizontal line just across columns 2 through 4, use \cline{2-4}.

Sometimes, you would like to have one item span several columns. You
can do this by using a \multicolumn command in your table data. This com-
mand takes the form \multicolumn{ct}{fmt}{item} where ct is a number
that says how many columns should be covered; fmt contains exactly one r,
l, or c to say how the item should be justified (and can also contain one or
more | characters to draw vertical lines); and item is the actual text of the
item.

The standard look of tables is a bit cramped vertically for my taste.
Adding the command \renewcommand{\arraystretch}{1.2} to the docu-
ment preamble will stretch arrays and tables vertically by a factor of 1.2,
giving them a less cramped appearance.

Note: To center a table on the page, put it in a center environment, that
is, between \begin{center} and \end{center}.

4.3 Graphics

It’s actually very easy to include graphics in your document, as long as you
use the graphics package as discussed above. This package makes available
the \includegraphics command, which takes the name of a graphics file
as a parameter. For example: \includegraphics{logो eps} Graphics files
can be in EPS (Encapsulated PostScript) format, and other formats such as
GIF or JPEG might also work. EPS is especially good for diagrams. Many
graphics and drawing programs can export images as EPS. If you want to
center an image on the page, put it into a center environment. Here, for
example, is the \LaTeX lion, scanned from Leslie Lamport’s original \LaTeX
User’s Guide and Reference Manual:

4.4 Cross References, Etc.

In a long paper, you will often want to say things like “See Subsection 2.5.”
This is called a cross-reference. If you actually type numbers like “2.5” into
the paper, though, you run into a problem when you add or delete sections,
since that will change the numbering. You would have to hunt down all the cross-references in your paper and change the numbers.

\LaTeX{} has a cross-referencing system that makes this unnecessary. The idea is to associate a label or name to each place that you might want to reference. You do this with the \verb|\label| command, which can immediately follow any command that uses automatic numbering, such as \verb|\section|. The parameter to \verb|\label| is a mnemonic name for the reference. \LaTeX{} will associate the actual number with this mnemonic name. For example, \verb|\section{Introduction}|\label{S-intro} associates the mnemonic name \texttt{S-intro} with the section number for a section named “Introduction.” Later, you can retrieve the number by using the \verb|\ref| command. This command takes the mnemonic name as a parameter. \LaTeX{} replaces the \verb|\ref| command with the actual number associated with the mnemonic name. In our example, \verb|\ref{S-intro}| would be replaced in the output with the section number of the “Introduction” section.

One fine point: When you use cross-references, you should process the \LaTeX{} input file twice. The first time allows \LaTeX{} to gather the reference numbers. The second time through, \LaTeX{} will insert the correct numbers in the correct places. (The numbers are actually stored in a separate file that \LaTeX{} creates automatically.)

\LaTeX{} is capable of helping with other bookkeeping chores as well. It can make a table of contents. It can number figures and tables and make cross-references to them. It provides a fairly easy way of creating an index for the document. And it has a bibliography environment that makes it easy to include citations to your references. If you are interested in any of this, it’s time to start looking for a more comprehensive \LaTeX{} reference...