```
%%%%%%%%%%%%%%%%%%%%%  place intro.tex here, carry out those edits
%%%%%%%%%%%%%%%%%%%%%  tex it, then adjust \spacingskip page by page as needed
\magnification=1200%\rightline{aug84; rev24jan90 04jan94, 06feb99}
\input format  % see page 8 for list of macros from format.tex  actually used
\pageno=5
\bigskip \centerline{Doing a \TeX job} \smallskip
\TeX\ is not a word processor. Rather, you use your favorite word processor or
file editor to prepare files of instructions for \TeX.
By and large, it doesn't matter what word processor you use. Since braces turn
out to be important in \TeX, you are better off using a file editor like
{\bf vi} that will allow you to check that every closing brace
(or parenthesis or bracket) matches the opening brace (or parenthesis or
bracket) you want it to match. It is also important that the word processor
leave no funny control characters in the file, as these might confuse \TeX.

How such an instruction file (traditionally carrying the suffix {\tt.tex}) is
converted into pages of beautiful print need not concern you (though you will
come to admire the cleverness that must have gone into the process). But you
have to find out how, on your computer system, you submit that instruction file
to \TeX. Usually, a command like  {\tt tex myfile} suffices if {\tt
myfile.tex} is the file containing the instructions. Your system should
also allow you to have the file worked on by \TeX\ without any final printing,
since that is the way to find out mistakes.

\bigskip                                   %<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
\centerline{\bf ordinary text}             %  ORDINARY TEXT
\smallskip                                 %<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<

You type ordinary text as you would in any word processor, except that there
are {\it fewer} things to pay attention to.
\item{(1)} As you can see,
it doesn't      matter    how many spaces    you leave between words
or after punctuation.     \TeX\ will choose an even and optimal spacing and
leave the right number of spaces after punctuation.
\item{(2)} As you can see,
it doesn't matter
where you
break lines. \TeX\ will choose optimal line breaks (if that is possible; it
will let you know when it cannot; actually, I tried hard to force such a
situation in the next item without trickery).
\item{(3)} In particular, you (usually) need not pay attention to hyphenation.
\TeX\ is completely familiar with all the rules and applies them automatically,
if need be. (If you are eager to supply a particular hyphenation, indicate the
place by typing {\tt\char92 -} there; e.g., {\tt surg\char92 -e\char'134-ry}.)

You indicate the end of a paragraph by leaving a blank line (or two or three if
you want to be generous; again, it doesn't matter how many you leave).
\par You can also indicate the beginning of a new paragraph by typing
{\tt \char'134 par}.

                                %<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
\def\ttb#1{{\tt\char'134 #1}}  %  convenient MACRO with one ARGUMENT
                                %<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
\bigskip
\centerline{\bf Special characters}
\smallskip
You type all characters great and small on your keyboard,
except that some (altogether ten) non-letters have
special meaning for \TeX\ and therefore must be typed in a special way if they
are actually to appear in the printed text. Here is the list of these special
characters:
\smallskip\centerline{
$\backslash$\quad$\{$\quad$\}$\quad\$\quad\&\quad\#\quad\^{}\quad\_\quad\%\quad\~{}}
\smallskip
\noindent You notice that you get most of these symbols to print by typing first
the backslash {\tt\char92}. For some, this will work only if this is (part of a
text) enclosed in
dollar signs (of which much more later). The backslash itself is so important
that \TeX\ requires you to type out the word {\tt backslash} after a
{\tt\char92} (and encase the whole thing in dollar signs) if you want to have
the backslash printed.

\vfill\eject
```

```
\advance\pageno by 1
The backslash serves as an {\bf escape character}, i.e., as a message to \TeX\
that what immediately follows the backslash is to be taken as a command (rather
than as text to be typeset).

In particular, you get all the symbols and signs {\it not} on your keyboard
with the aid of such {\bf command}s. See Sam Bent’s \TeX\ Reference Card in the
Appendix for a complete, ordered listing.

\heading{odds and ends}
\TeX\ will automatically leave more space after a period at the end of a
sentence than between words in a sentence. But since \TeX\ has no way of
understanding what you write, it has to make a guess at what is ‘a period at
the end of a sentence’. It guesses that it is any period followed by one or
more blanks as long as it is not preceded by a capital letter, as in D. E.
Knuth. Note that \TeX\ provides an ordinary interword space after each of
the two initials, and a larger, intersentence, space after the final
period.

This rule of thumb causes difficulty when a sentence ends in a capital
letter, such as this ONE.
See? You can overcome it by inserting the {\bf do nothing} command
\ttb{null} just prior to the PERIOD\null. There. That does it.

This rule of thumb also causes difficulty with abbreviations, such as refs. to
comp.lit. or phys.ed. courses. You overcome this by following such
abbreviation periods by a {\bf forced blank}, e.g., as in refs.\ to
comp.lit.\ or phys.ed.\ courses. See the difference?

This rule of thumb causes difficulty with the unhappy habit and unfortunate
standard rule of ‘putting the period at the end of a quote {\it inside}
the quote.’ The remedy is simple: ’Put the period outside’. But if you must keep
it inside, ‘‘follow the quote with two forced blanks.’’\ \ There.

Note that the beginning quotes in the preceding paragraph were
typed differently from what you might have expected. Just to get used to
this, also type the closing double quote as ’’ even though " will
give the same thing.

\heading{simple math}
All math within text is enclosed within dollar signs, even when it is just one
letter or symbol, such as $a$, $b$, or $c$, and certainly for things like
$E=mc^2$, or $x_i+y_i={3\over4}$, or $a^{b^c}$, or $x^\alpha_1$, or
or $\int_{-1}^1x\,dx=0$, or
$\sum_{i=0}^nc^i=(c^{n+1}-1)/(c-1)$ for all $n\ge0$.

It is worth studying these examples in some detail. I have not left any spaces
around symbols; \TeX\ takes care of that, usually. But \TeX\ cannot understand
math, so sometimes you may have to control spacing, as I have done in the
integral, by putting a bit of space (via {\tt\char’134,}) between the $x$ and the
$dx$. Also, one must use braces to indicate the extent of subscripts and
superscripts, but only if they involve more than one character. Braces are also
required to control what \ttb{over} puts on top and below the bar.

For a complete, ordered list of all the available math symbols, see Sam Bent’s
\TeX\ Reference Card in the appendix.


The inexperienced typist will bemoan having to type all those dollar
signs. Although I do touch-typing, sort of, I have found it more convenient to
type the dollar sign (and some other signs) piano-fashion, i.e., by hitting
appropriate keys simultaneously with the\hfill\break\vfill\eject
```

```
\advance\pageno by 1
\noindent same hand; in this example, it's the
left hand, with the little finger hitting the shift key while the middle finger
simultaneously hits the 4/\$ key.

\heading{display math}
Displayed math is enclosed within {\bf double} dollar signs. Here are some
of the earlier math examples, but in double dollar signs:
$$
E=mc^2, or x_i+y_i={3\over4}, or a^{b^c}, or x^\alpha_1,
$$ or
$$\int_{-1}^1x\,dx=0,$$ or
$$\sum_{i=0}^nc^i=(c^{n+1}-1)/(c-1), for all n\ge0.$$
You'll notice that some of the display has become more expansive. For
example, the summation sign is bigger, and its limits are above and below it,
rather than to the right of it. That's nice since \TeX\ takes care of such
things; you type it the same way, whether in math mode or in display mode,
and \TeX\ makes the appropriate adjustments. You'll also notice that the
last comma in the first and second display, and the period in the third
display, are now
typed before the dollar sign(s) rather than after. Can you guess what would
have happened otherwise?

On the other hand, notice what happened to the ordinary word ``or'' and the
text ``for all''. \TeX\ treats them, not as words, but as a sequence of math
symbols, hence puts them into italics, gives them a funny spacing and ignores
entirely the interword space. To have them typeset as ordinary text, you
need to switch temporarily to that font, as in the following:
$$\sum_{i=0}^nc^i=(c^{n+1}-1)/(c-1), {\rm for all} n\ge0.$$
This now uses ordinary (roman) type for these two words, but still isn't
right since, being inside display mode, the interword spacing is ignored.
So you have to enforce it by using the {\bf forced blank}:
$$\sum_{i=0}^nc^i=(c^{n+1}-1)/(c-1), {\rm for\ all\ } n\ge0.$$
This is still not quite right since it is nicer to set off that last
quantifier from the actual formula. This you do by inserting some
(standard) space that \TeX\ provides with the commands \ttb{quad}
and \ttb{qquad}. \ttb{qquad} is the right one here:
$$\sum_{i=0}^nc^i=(c^{n+1}-1)/(c-1),\qquad {\rm for\ all\ } n\ge0.$$

\vfill\eject
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   here, for the record, is the list of the macros from the file
%    format.tex  (input initially) that are actually used here.
%
%

\def\heading#1{\bigskip\centerline{#1}\smallskip}
\def\sub{_{\bf|}}
\def\openZ{{{\rm Z}\kern-.28em{\rm Z}}}
\def\ga{\alpha}
\def\gO{\Omega}
\def\strutdepth{\dp\strutbox}
\def\marginal#1{\strut\vadjust{\kern-\strutdepth%
\vtop to \strutdepth{
    \baselineskip\strutdepth
    \vss\llap{\fiverm#1\ }\null}}}
\def\inpro#1{\langle#1\rangle}
\def\emember#1{\ifrememberchapter % subsequent percents prevent unwanted blanks
\immediate\write\chpaux{\def\string#1\chapterno{\chapterno.\formeq}}\fi%
\immediate\write\aux{\def\string#1{\formeq}}%
\global\edef#1{\formeq}}
\newif\ifrememberchapter\newread\testfl
\def\inputifthere#1{\immediate\openin\testfl=#1
\ifeof\testfl\message{(#1 does not yet exist)}\else\input#1\fi\closein\testfl}
\inputifthere{\jobname.aux}\newwrite\aux\immediate\openout\aux=\jobname.aux


%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
\advance\pageno by 1

\heading{alignment}
\TeX\ really begins to pay off when it comes to the alignment of text. Here
are some standard {\it examples}:
$$
H(x):=\cases{x&if $0\le x\le 1$;\cr
             2-x&if $1\le x\le2$;\cr
             0&otherwise.\cr}
$$
This is an example of the \ttb{cases} statement. The description
of each line has two {\bf
fields}. The first field goes from the ‘beginning’ of the description to the
{\bf ampersand} {\tt\&}; it is typeset automatically as {\it math}. The second field
goes from the ampersand to the {\bf carriage return} \ttb{cr}, and
it is typeset automatically as {\it text}; hence any math in it must be
embraced by dollar signs. The description of the first line begins right at
the opening brace of the \ttb{cases} statement. The description of
subsequent lines begins right after the \ttb{cr} of the preceding
line. (It doesn’t matter how many actual lines you use in your
{\tt.tex} file to describe the lines to be displayed in the cases
statement.)

If you prefer somewhat more space between the cases, use {\tt\char’134
noalign}, as in this modification, which also shows how you could get the
material in the first fields centered:
$$
H(x):=\cases{\hfil x\hfil&if $0\le x\le 1$;\cr\noalign{\vskip6pt}
             2-x&if $1\le x\le2$;\cr\noalign{\vskip6pt}
             \hfil0\hfil&otherwise.\cr}
$$
The next example shows how to align several lines of displayed equations.
The description of each line is terminated again by the carriage return,
and, in each description, the point of alignment is again marked by an
ampersand.
$$\eqalign{M{*}’f&=M{*}(M\sub{*}c)\cr
&=M{*}(c{*}M\sub)\cr
&=(M{*}c){*}M\sub=f{*}’M.\cr}$$
Incidentally, the asterisks have all been em‘brace’d here to prevent \TeX\
from embedding them in extra space (a thing it would do to any symbol it
recognizes as a binary operation sign).

Many math alignments can be handled by using {\bf matrices}. Here is the
standard example:
$$
A=\pmatrix{a_{11}&a_{12}&\ldots&a_{1n}\cr
           a_{21}&a_{22}&\ldots&a_{2n}\cr
           \vdots&\vdots&\ddots&\vdots\cr
           a_{m1}&a_{m2}&\ldots&a_{mn}\cr}
\leqno{(14)}$$
in which you also see various ways of entering the equivalent of {\bf etc} into
math formulas as well as an equation number.
The macro \ttb{pmatrix} provides its own fences. If you prefer some other kind,
you use \ttb{matrix} together with \ttb{left} and \ttb{right}, as in
$$\left[\matrix{\rm A&\cal B&\it C\cr
                \bf D&\sl F&\Gamma\cr}\right],\eqno{(14)’}$$
\vfill\eject
```

```
\advance\pageno by 1\noindent which
also illustrates that you can treat each matrix entry as if it were
typeset without regards to any other entry (which it is).

Note that, once again, the ampersand {\tt\&} indicates the alignment points
while the carriage return \ttb{cr} indicates the end of a line (description).

The use of {\bf over-} and {\bf underbrace} is illustrated nicely here:
$$
\{\underbrace{\overbrace{a,\ldots,a}^{h\;a\rm's},
              \overbrace{b,\ldots,b}^{k\;b\rm's}}_{h+k\;\rm elements}\}.
$$
Actually, this looks a bit crooked. To get it right, add a \ttb{mathstrut}
to each group to be ‘overbraced’ in order to give them the same height:
$$
\{\underbrace{\overbrace{\mathstrut a,\ldots,a}^{h\;a\rm’s},
              \overbrace{\mathstrut b,\ldots,b}^{k\;b\rm’s}}_{h+k\;
                                                           \rm elements}\}.
$$
This is still not quite right, since we would want apostrophes rather than
primes in the overbrace material, but ’twill serve.

%\halign forces switch to vertical mode if used in horizontal mode<<<<<<<<<<<<
The next example shows the general-purpose and very useful \ttb{halign}
command in action:
\halign{#&\qquad$#$&\quad$#$\quad&\hfil$#$\hfil&$\;#\;$&\hfil$#$\hfil\cr
for $f\in\Pi_M$,&M{*}’f=f{*}’M&=&
f\sum_{j\in\openZ^d}M(j)&-&\sum_{j\in\openZ^d}(f-f(\cdot-j))M(j)\cr
&&\in&f&+&\Pi_{<{\rm deg}f}\cr}
Here, too, we describe how each of the displayed lines should look.
As before, {\bf field}s are separated by ampersands
{\tt\&}’s, and line descriptions by carriage returns \ttb{cr}’s.
But the first line description is special. It doesn’t describe a particular
line; rather, it is a {\bf template} for the actual lines, i.e., it
describes a general {\bf pattern} to be followed by all
(subsequent) lines. This means that each field in the first line
description tells how the {\bf corresponding} field in the actual lines
should be handled, with the {\bf sharp} sign {\tt\#} indicating just where
in that field the actual material in each line for that field is to
appear.

In the above example, the description of the first field is simplest. It
contains the sharp sign {\tt\#} and nothing else. The second field starts with some
space and then has the sharp {\tt\#} embraced by dollar signs. Hence, for each
line, the second field will start with the same specified amount of space followed by
the actual material from the second field, but typeset in {\it math}
mode. The fourth field (it is part of the trickiness of {\tt\char’134
halign} to keep proper track of which field is which) uses {\tt\char’134
hfil} to put some {\bf variable} space around the material for that field,
which, in effect, {\bf centers} that material within the field. It also
embraces the material with dollar-signs, hence will do it in math mode. The
next field is entirely in math-mode, but embraces the sharp sign with a
little space (indicated by {\tt\char’134;} which only works in math-mode),
and so puts that space around the material for that field in each actual
line. And so on.

The only thing not quite right about the above display is lack of centering and lack of
proper surrounding space. But that can be arranged, using {\tt\char’134
noalign} to put some space between the lines, and things called
\ttb{hbox} and \ttb{vbox} to center it all, as you see here:
\vfill\eject
```

```
\advance\pageno by 1
\def\sumZ{\sum_{j\in\openZ^d}}   %    <<<<   locally useful abbreviation
\bigskip\hbox to \hsize{\hfill\vbox{
\halign{#&\qquad$#$&\quad$#$\quad&\hfil$#$\hfil&$\;#\;$&\hfil$#$\hfil\cr
for $f\in\Pi_M$,&M{*}'f=f{*}'M&=& f\sumZ M(j)&-&\sumZ (f-f(\cdot-j))M(j)\cr
\noalign{\medskip}
&&\in&f&+&\Pi_{<{\rm deg}f}\cr}}
\hfill}\bigskip
I have to admit that getting an alignment of such complexity right takes
some doing and some patience, and some struggle with \TeX's often
inscrutable error messages.

This particular example might actually be easier to handle with the
\ttb{matrix} command, which gives:
$$\matrix{
\hbox{for\ }f\in\Pi_M,\ \ &M{*}'f=f{*}'M&=&f\sumZ M(j)&-&\sumZ\big(f-f
(\cdot-j)\big)M(j)\cr \cr &&\in&f&+&\Pi_{<{\rm deg}f}\cr}$$

Here is a simple use of \ttb{halign} which also demonstrates the use of
\ttb{hbox} and \ttb{vbox}, well, of its cousin \ttb{vtop} (see the section
on boxes for details).
\medskip
\hbox to\hsize{\hfill\vtop{
\halign{#\hfill\cr Michael G. Crandall\cr Department of
Mathematics\cr University of Wisconsin-Madison\cr Madison WI 53706\cr}}
\quad and \quad \vtop{
\halign{#\hfill\cr Pierre-Louis Lions\cr Ceremade\cr
Universit\'e Paris-Dauphine\cr Place de Lattre de Tassigny\cr
75775 Paris Cedex 16\cr}}\hfill}
\medskip
The function served here by \ttb{halign} is to permit the
name-and-address information to be entered in line, separated only by the
carriage returns, and to make sure that all these lines are left-adjusted,
and to supply a box that is exactly large enough to contain all that
information, hence makes it possible to align these two boxes properly on
the page. But it makes it also very easy to {\bf center} the information
instead, by adding another \ttb{hfill} to the template in \ttb{halign}:
\medskip
\hbox to\hsize{\hfill\vtop{
\halign{\hfill#\hfill\cr Michael G. Crandall\cr Department of
Mathematics\cr University of Wisconsin-Madison\cr Madison WI 53706\cr}}
\quad and \quad \vtop{
\halign{\hfill#\hfill\cr Pierre-Louis Lions\cr Ceremade\cr
Universit\'e Paris-Dauphine\cr Place de Lattre de Tassigny\cr
75775 Paris Cedex 16\cr}}\hfill}
\medskip

Table construction is also easily done with the aid of \ttb{halign}.
Here is a simple {\bf table}.
\bigskip \centerline{\bf Table 1}
\medskip
\hbox to \hsize{\hfill
\vbox{\halign{\hfill#\hfill&\vrule#&\hfill#\hfill\cr
$x$&&$\sin(x)$\cr \noalign{\hrule}
0&&0\cr
$\pi/2$&&1\cr}}\hfill} \bigskip
Note the use of rules to draw lines. Note in particular how that vertical
line is drawn for each line with the aid of the second field in the
pattern. Note that the pattern description for that second field must
contain a sharp even though you have no intention of ever putting anything
into that field (other than the \ttb{vrule} specified by the pattern).

But it's a miserable table otherwise. The heading line and the first line
in the table are right on top of each other, yet there's some skip between the
two lines in the table, and the columns are just big enough to contain the
widest item in each.
\vfill\eject
```

```
\advance\pageno by 1

We fix the crowding of the lines by putting a \ttb{strut} into the pattern
(i.e., a \ttb{vrule} of zero width but of sufficient height and depth that
the resulting lines of type can be allowed to be contiguous without any
interlineskip and still look well separated). This will also fix the problem
with the skip between the lines; for, with the \ttb{strut} making all lines
a little bit larger, we can turn off the interlineskip inside the box that
is to contain the table by saying \ttb{offinterlineskip}. Finally, we can
make the columns a little bit wider by adding some forced blanks in the
patterns. Here is the result:
\bigskip \centerline{\bf Table 1} \medskip
\hbox to \hsize{\hfill \vbox{\offinterlineskip
\halign{\strut\ \hfill#\hfill\ &\vrule#&\ \hfill#\hfill\ \cr
$x$&&$\sin(x)$\cr \noalign{\hrule} 0&&0\cr $\pi/2$&&1\cr}} \hfill}
\bigskip

For the more demanding consumer, here is a more sophisticated table in
which repeated \ttb{hrule}s have given some lines greater thickness, and in
which some entries actually go across several columns.
\bigskip \centerline{\bf Table 2} \medskip
\hbox to \hsize{\hfill\vbox{\offinterlineskip
\halign{\strut\vrule\vrule\vrule#&
\quad\hfill#\hfill\quad&\vrule#&
\quad\hfill#\hfill\quad&\vrule#&
\quad\hfill#\hfill\quad&\vrule\vrule#&
\quad\hfill#\hfill\quad&\vrule#&
\quad\hfill#\hfill\quad\vrule&\vrule#&
\quad\hfill#\hfill\quad&\vrule#&
\quad\hfill#\hfill\quad&\vrule\vrule\vrule#\cr
\noalign{\hrule\hrule\hrule}
&\multispan5 \hfil$\theta=2\deg$\hfil&&\multispan3\hfil Flight data \hfil&&
\multispan3\hfil Computation\hfil&\cr \noalign{\hrule\hrule}
&$R^*$&&$\tau^*$&&$b$&&$C_{lrm}$&&$C_{lsm}$&&$C_{lrm}$&&$C_{lsm}$&\cr
\noalign{\hrule\hrule\hrule}
&10.0&&.091&&4.32&&-.025&&.020&&-.02375&&.02437&\cr \noalign{\hrule}
&20.0&&.087&&5.20&&-.040&&.040&&-.02984&&.03571&\cr \noalign{\hrule}
&45.2&&.123&&4.23&&-.055&&.050&&-.04454&&.04714&\cr \noalign{\hrule\hrule\hrule}
}}\hfill}
\bigskip
\noindent Note the use of \ttb{hbox to \char'134hsize} and \ttb{hfill} to center
the table. By the way, in typing in the description, I made extensive use
of the fact that, in {\bf vi}, it is very easy to duplicate and repeatedly
use a word, or phrase, or entire line, and that {\bf vi} will automatically
show me matching braces, so I would be sure which grouping I was closing.
As a result, the table came out perfect the first time I ran \TeX\ on it,
-- except that I had typed {\tt Fight} rather than {\tt Flight}, and had typed
{\tt C\char'137\char'173\char'134rm\char'175} rather than
{\tt C\char'137\char'173lrm\char'175}. Immerhin.

\def\tbb#1{{\tt\char'173 #1\char'175}}%%<<<<<<<<<<<<

\TeX\ also makes available the main tool for vertical alignment on a
typewriter, the {\bf tab key}. I have never used it since I find \ttb{halign}
so much more versatile and I don't have to count spaces beforehand. (In fact,
\TeX\ implements tabbing with the aid of \ttb{halign}.) Tabbing is useful for
horizontal alignments that run over many pages since, in contrast to
\ttb{halign}, tabbed material can be typeset line by line. By contrast,
\ttb{halign} must first look at all the columns before it can settle column
width and spaces between columns.

Finally, there are occasions when you want to locate material just so. For
example, you might have a graph you wish to label. (\TeX\ makes available
the macro \ttb{special} which allows you to include, e.g., graphic material
right with the \TeX ed material. Details depend on the printer you are
using and on the particular software that converts the dvi file \TeX\
produces into a file that makes sense to your printer.) For this, look for
the macros \ttb{gridbox} and \ttb{point} in the section headed `including and
labelling figures', which are also useful for precise placement without any
figure involved.

We now use \ttb{vfill\char'134 eject} to generate a clean page break.
\vfill\eject % a PAGE BREAK has been forced here <<<<<<<<<<<<<<<<<<<
```

```
\advance\pageno by 1
\hoffset=2cm\voffset=-1cm
{\hsize=15truecm\vsize=24truecm
\heading{page layout}
The size of the overall area on a page to be filled with print can be
controlled by setting \ttb{hsize} and \ttb{vsize}.

The placement of this rectangle of printed text is
controlled by setting \ttb{hoffset} and \ttb{voffset}.

{
\parskip=10pt The space between paragraphs is
controlled by \ttb{parskip}.

\parindent=3cm The amount of indentation is
controlled by \ttb{parindent}.

\baselineskip=18pt
The amount of space between lines can be
controlled by setting \ttb{baselineskip}, as I have just done. All
these parameters have default values. You set them only to change those default
values. Their new values are used as soon as you set them and until you come to
the end of the current grouping. After that, they revert to what they were
before entering that grouping.\par
}


{\narrower\indent % will increase the margins on either side of a grouping
If you want to restrict these changes to a particular part of the text, put that
text into braces that also embrace those changes, as I have done for the
preceding three paragraphs (and also for the entire material on this page).
\par}
{\narrower\narrower\noindent The preceding paragraph has had its margins
widened by use of \ttb{narrower}. Also, the indentation for the present
paragraph has been suppressed by a \ttb{noindent}, and the paragraph
made yet narrower by a \ttb{narrower\char'134 narrower}. But, in
each case, the effect of the narrowing command has been restricted to the
paragraph by embracing the {\it completed} paragraph.\par}

It is important to note that parameters that only affect entire
paragraphs (such as \ttb{baselineskip} or \ttb{narrrower})
only affect those paragraphs that are completed before the end of the grouping
within which they occur is reached. For {\it example}, leaving off the
\ttb{par} near the end of the description of the preceding
paragraph would have prevented the \ttb{narrower\char'134
narrower} at its beginning from taking effect.

\parindent=5pt
\hang \ttb{parindent} has just been made quite small. Also,
\ttb{hang} has been used to indent the entire paragraph. The amount of
indenting is determined by the current value of \ttb{parindent}.

\item{$\bullet$} You can get the same effect with \ttb{item}, except that
\ttb{item} gives you the opportunity to put something to the left
of the indent on the first line, as I have done here.

Note the effects of

\leftline{a \ttb{leftline}}
\centerline{a \ttb{centerline}}
\rightline{and a \ttb{rightline}}
\line{and of a \ttb{line}\hfill with \hfill some\hfill  \ttb{hfill}}
}
\vfill\eject
```

```
\advance\pageno by 1
\hoffset=0pt\voffset=0pt  %<<<<<<<<<<<<<<<<<<<<<  restore standard placement

\font\hfoo=cmr7 %scaled \magstephalf
\font\foo=cmbx10
\headline={\ifodd\pageno \rightheadline \else \leftheadline \fi}

\def\rightheadline{
        \ifnum\pageno<1\hfoo\hfil a sample headline\hfil\number\pageno
        \else
             \ifnum\pageno=1 \hfil\hfoo \folio
             \else \hfoo \hfil \chapterno.\ \chaptername \hfil \folio
             \fi
        \fi}
\def\leftheadline{\ifnum\pageno<1 \hfoo a sample headline\hfil\number\pageno
                  \else \hfoo \hfil \sectionname \hfil \folio\fi}
\footline={\hfoo\ifnum\pageno<0 \hfil\folio\hfil
              \else\hfil a footline!\fi}
\def\chapterno{3}\def\chaptername{Headlines!}
On this page, you see some heading and some footing. Also, the page number
also occurs at the upper right corner. All
of this is the result of setting two built-in macros called \ttb{headline}
and \ttb{footline}. As you read the definitions, you will appreciate the
possibility of having the left-page headline differ from the right-page
headline, or of having the page number appear at the bottom center as long
as it is a roman numeral (indicated by having \ttb{pageno} negative), and
at the outer upper page corner otherwise.

For this, you need to know that \ttb{folio} gives you the decimal digits of the
current page number (i.e., the value of \ttb{pageno}) in the current font,
in case that number is nonnegative, and gives you the roman numerals of the
negative of \ttb{pageno} otherwise. That works out fine for the title matter
in a book (and since the Romans didn't have a zero). You also want to know
that \ttb{pageno} is something that \TeX\ will increase by one every time it
finishes a page (or decrease by one if it was negative). But, if you don't
like the current number and rather have it be the number 1937, you could
say \ttb{pageno=1937} and that would take care of it.

In fact, if you don't want page numbers at all, say \ttb{nopagenumbers}.
This is the same as saying \ttb{footline\tbb{\ttb{hfil}}}, but makes the
purpose clearer.
\newcount\savepageno\savepageno=\pageno\pageno=-3% equal signs used for readability

I'll restore all of this (including \ttb{pageno}) back to what it was at the
beginning of the next page. Look there for how it is done.

More complicated schemes are available. E.g., it is possible to have the
footline contain the latest marked expression in the text prior to the page
break. But for such things, you had better consult the book.

It is also good to know how to do a
\bigskip\centerline{Table of Contents} \medskip
\def\leaderfill{\leaders\hbox to 1em{\hss.\hss}\hfill}
\def\leaderarrow{\leaders\hbox spread 1em{--$>$}\hfill}
\line{Disclaimer and Exhortation\leaderfill1}
\line{A page of \TeX\leaderarrow3}
\line{Doing a \TeX job\leaderfill5}
\def\goon#1#2{\line{#1\leaderfill#2}}
\goon{ordinary text}5
\goon{Special characters}5
\goon{odds and ends}7
\goon{simple math}7
\goon{display math}9
\goon{alignment}{11}
\line{tables\leaderarrow15}
\goon{page layout}{19}
\goon{Table of Contents}{\folio}
\goon{commands, macros, definitions}{23}
\goon{commands with arguments}{23}
\goon{indexing and labelling figures}{27}
\goon{sequencing equations and other items}29
\goon{boxes}{30}
\goon{errors}{33}
\vfill\eject
```

```
\headline={}\footline={\hfil\folio\hfil}
\pageno=\savepageno\advance\pageno by 2

\def\hanginthere{you are doing fine!}
\bigskip \centerline{{\bf commands}, macros, definitions} \smallskip
\TeX\ commands (also called {\bf macro}s, or {\bf definition}s) begin with a
backslash and are followed either by exactly one non-letter, or else by one or
more letters.  There is no limitation on how many letters such a command `word'
is made up from. \TeX\ will read the letters until it comes to a non-letter and
then take all the letters read together as being the command word. If that
terminating non-letter is a blank (or space), then it (and all blanks
following it) will be lost.

For {\it example}, you have been seeing the \TeX\ logo. The instruction for its
typing is such a command, but you see the command being followed by another
backslash and a space. The latter command is of the non-letter variety, the
non-letter being the blank. It tells \TeX to put a blank space in. Can you tell
why that is necessary? Look also for earlier examples, in which the command for
the logo is {\it not} followed by the blank command.

Available commands are either {\bf primitive} commands, or else {\bf macros},
i.e., an {\it abbreviation} for a sequence of commands. The description of the
heading for the next section
contains an {\it example} of such a macro, defined just prior to it to combine
the three commands used in the description of earlier headings.

Sam Bent's \TeX\ Reference Card in the Appendix contains a complete, ordered
listing of all the primitive commands as well as the plain\TeX\-supplied macros.
But the fact that you can make up your own macros to suit your own purposes
is one of the major attractions of \TeX.

For {\it example}, I have collected a file consisting entirely of macro
definitions. The file happens to be called {\tt format.tex}.
To make certain that these macros are available to me when I work on a {\tt.tex}
file, I start the {\tt.tex} file with the command \ttb{input format}.  This
instructs \TeX\  to become familiar with all those macros in the file {\tt
format.tex} before starting to work on the instructions in the {\tt .tex} file.
The full file is available by anonymous ftp at {\tt ftp.cs.wisc.edu/Approx} and
includes definitions like

{\def\bs{\char'134}\def\lb{\char'173}\def\rb{\char'175}

{\tt\bs def\bs ga\lb\bs alpha\rb} for {\bf g}reek {\bf a}lpha; or

{\tt\bs def\bs gO\lb\bs Omega\rb}
for {\bf g}reek Capital {\bf O}mega, so I only have to type {\tt\$\bs ga\$
and \$\bs gO\$} to get `$\ga$ and $\gO$'; or

{\tt\bs def\bs bs\lb\bs backslash\rb}
for the printed {\bf b}ack{\bf s}slash, so I only have to type \ttb{bs} to get a
backslash in the text (instead of
the long word {\tt backslash}).
\par (Please admire the actual instructions for the typing of the
last item. The great convenience brought on by making ten characters special
brings much inconvience when you actually want to print those characters.
(The difficulty is compounded by writing it all in typewriter font.)
Fortunately, you seldom need to type these special characters.
My trip-up is usually the dollar or percent sign in a letter, or the sharp
sign in math-mode (where it denotes cardinality), or the ampersand in a list
of names or a reference.)
}

Why would use of the macro \ttb{deadly} defined by
\ttb{def\ttb{deadly}\tbb{\ttb{deadly}}} be deadly?
                                                %<<<<<<<<<<<<<<<<<<<<<<<<<<<
\def\heading#1{\bigskip\centerline{#1}\smallskip} %NOTE this example of a macro
                                                % with an argument.
                                                %<<<<<<<<<<<<<<<<<<<<<<<<<<<
\heading{\bf commands with arguments}
The heading of this section is put in with the aid of the
macro \ttb{heading}.  Have a look at it.  It uses an argument, namely
the material that is to appear in the heading. Now look at its definition,
just above its use. You can tell from the definition that it is intended
to
\vfill\eject%<<<<<<<<< another page break put in to have left and right pages match
```

```
\advance\pageno by 1
\noindent use an argument because you see the symbols {\tt\#1} right between the name
of the macro and the opening brace of its ‘body’. Also, you see within the body
the symbols {\tt\#1} repeated right at the spot where the material that
makes up the argument is to be placed.

Here is how such a macro is understood by \TeX. After \TeX\ has read the
macro’s name (\TeX\ will know that the name is {\tt heading} and not
something longer because the character following the {\tt g} in
\ttb{heading} either is a blank or else a non-letter), \TeX\ will look up the
definition, find that it requires an argument and now look for it. It
expects to find an opening brace as the next character. If that is indeed
the case, it will take everything between this opening brace and the {\it
corresponding} closing brace as the argument. In this particular example,
this means that everything between that opening and corresponding closing
brace will end up centered, with the centerline preceded and followed by
some vertical space.

\TeX\ also allows for a shortcut. If an argument consists of just one
character, then it is not necessary to enclose it in braces. In other
words, if \TeX\ does not find an opening brace as the next character, then
it takes that next character as the whole argument.

A macro may have up to nine separate arguments. In the definition of the
macro, they are listed, between the macro’s name and the opening brace of
the macro’s ‘body’, as {\tt\#1\#2...}. They should also (but don’t have to)
appear at least once inside the macro’s body, exactly at the spot at which the
material that forms the argument is to appear.  The intent is to have the macro
provide a template with certain places left open, to be filled in with
particulars when the macro is actually used.

At first glance, this list of {\tt\#1\#2\#3...} between macro name and
body looks a bit silly; why not simply say {\tt 5} if five arguments are
expected? But this is yet another cleverness of \TeX. For, you are
permitted to put any one character after each of those numbers, for
example, {\tt \#1,\#2|\#3\#4/...}, and these very characters are used later
by \TeX\ to decide when one argument ends and the next one begins. For
example, I have a friend (not a piano player) who hates typing dollar
signs. He has a macro that he defined as follows:

\ttb{def\ttb{m\#1:\tbb{\$\#1\$}}}

\noindent If he has to type something in math-mode, e.g. $\alpha=1/\gamma$,
he would type

\ttb{m\ttb{alpha=1/\char’134gamma}:}

\noindent \TeX\ will pick up the {\tt m} as the
name of a macro, look it up and find that it has one argument and that the
extent of the argument is all the stuff following that \ttb{m} until it
comes across a colon. So it picks up all that stuff and, following
instructions, puts it between dollar signs and then processes it in the
usual way. The colon itself will not be printed; it was used up as the
{\bf delimiter} or {\bf terminator}  of the first (and only) argument of the macro.

What is he going to do when his math-stuff contains a colon, e.g., he wants
to type $\{x: f(x)=0\}$? Then he hides that colon in braces! I.e., he types

\ttb{m\char’134\char’173x\tbb{:} f(x)=0\char’134\char’175:}

\noindent For, as \TeX\
reads the stuff following \ttb{m}, it skips over any groupings, i.e. over
any stuff between braces in its search for the delimiting or terminating
colon.

\vfill\eject
```

```
\advance\pageno by 1
\beginsection{including and labelling figures}

Here are the definitions of \ttb{gridbox} and \ttb{point}. These macros are
useful for the precise placement of material. I illustrate their use in
labelling a simple graph. The actual graph is specified with the aid of
PostScript, the same language that is used on many Laser writers to print
\TeX. Different printers or different converters from dvi-file to printer
file would require different statements to get the graphic combined with
text. A popular (free) means for placing figures into \TeX\ text is \ttb{epsf}.

\def\tick#1/{\vrule width\gridwidth height0pt depth#1truecm}
\def\nexttick#1#2/{\hbox to #1truecm{\hfil\tick#2/}}
\newcount\tickcount
\newdimen\gridwidth \gridwidth=0.4pt
\newif\ifshowgrid \showgridtrue
\def\point(#1,#2)#3{\vbox to 0pt{\kern#1truecm
   \hbox{\kern#2truecm{#3}}\vss}\nointerlineskip}

\def\gridbox#1/#2/#3{
\vbox to #1truecm{
#3
\ifshowgrid\tickcount=0
  \loop
   \vbox to 0pt{\kern\tickcount truecm\hrule width#2truecm height\gridwidth\vss}
   \nointerlineskip \advance\tickcount by 1
   \ifdim\tickcount pt<#1pt\repeat % Note: #1, #2 need not be integers
  \hbox to 0pt{\tickcount=1\tick#1/
  \loop\ifdim\tickcount pt<#2pt\nexttick1 #1/\advance\tickcount by 1\repeat\hss}
\else \vbox to 0pt{\hrule width#2truecm height0pt\vss}
\fi\vfil}\vfil}

We start with the graphic, put into the gridbox, with the grid drawn, to
help us later on to place the labels.
\bigskip\hbox{
    \vbox{ \gridbox3.5/6.6/{\point(0,0){\special{ps: plotfile intsamp.pls}}
        }} }\bigskip

Next we take a stab at placing the labels.
\bigskip\hbox{
    \vbox{ \gridbox3.5/6.6/{\point(0,0){\special{ps: plotfile intsamp.pls}}
       \point(3,.5){$A$}\point(3.2,6.1){$C$}\point(.2,2.45){$R$} %<<<<<< LABELS
        }} }\bigskip

That looks ok, except that the $R$ should be a little bit more to right and
down, say .35cm down and .4 cm to the right. So we change these
`coordinates'. We also get rid of the grid (which we can do by saying
\ttb{showgridfalse} or by saying \ttb{gridwidth=0pt}), center the whole
figure, and put a Figure description underneath.
\showgridfalse
\bigskip\hbox to \hsize{
    \hfil\vbox{ \gridbox3.5/6.6/{\point(0,0){\special{ps: plotfile intsamp.pls}}
       \point(3,.5){$A$}\point(3.2,6.1){$C$}\point(.55,2.9){$R$}%<<<<<< LABELS
             }}\hfil }
\centerline{Figure 3. A labelled graph} \bigskip
There, that wasn't so bad. -- One would usually work on this in a separate
file and only insert the finished material appropriately, perhaps using
\ttb{midinsert} or \ttb{topinsert}.

\vfill\eject

% Here is the plotfile intsamp.pls used on this page. you'd have to remove
% the leading percent signs before using it.

%% save
%% % 23.625 23.625 scale% this is the right scale (with dvilaser) to make the
%%                      %   given (x,y) exactly cm when using \magstep=1200.
%% 236.25 -236.25 scale % is the right scale for dvips.
%% currentpoint translate
%% .02 setlinewidth 1 setlinejoin 1 setlinecap
%% newpath 1 -3 moveto 3 -1 lineto 6 -3.5 lineto stroke
%% restore
```

```
\advance\pageno by 1

\def\boxit#1{$$\vbox{\hrule\hbox{\vrule\kern10pt
     \vbox{\kern10pt\vbox{\hsize 15.6truecm \noindent \strut
#1\strut}\kern10pt}\kern10pt\vrule}\hrule}$$}

\boxit{At times, it is convenient to pack it all in. This is one of those
times, I think. (You might try to modify the macro \ttb{boxit} used here to
get a double box line.)}

\def\keyword#1{$\inpro{#1}$}
\beginsection{sequencing equations and other items}

\newcount\equationno \equationno=0
\def\formeq{\the\equationno}

\def\label#1/{\elabel{#1}///}\newif\ifshowlabel
\def\elabel#1/#2/#3/{\global\advance\equationno by 1 %
\ifx#1\empty\else\emember#1%
\ifshowlabel\marginal{\string#1}\fi\fi%
\ifmmode\leqno{#3(\formeq#2)}\else#3\formeq#2\fi} %<------ switch to \eqno ???

(\label/) Here is the macro \ttb{label}
which is useful for an automatic sequencing of equations
and other items. It is so simple that it wouldn't be hard for you to
modify them to fit the particular needs of a particular paper.

(\label\listequ/) \ttb{label} will increment the value of \ttb{equationno} by
one and then print it out, and will enclose it in parentheses if called in
display-math mode, as in the following.   $$e=mc^{1/2}\label/$$

(\label/) In order to refer to these numbers later, you give them names.
For example, I gave the name \ttb{listequ} to the number that starts the
paragraph (\listequ) ({\bf check how I typed the preceding number})
by saying \ttb{label}\ttb{listequ} there.
\equationno=37

(\label/) You can always change \ttb{equationno} to any value you like. For
example, I just changed it to 37 by typing \ttb{equationno=37} prior to this
paragraph.

\showlabeltrue
(\label\memory/) If you find that you have trouble remembering the names
you gave to earlier items (and you don't want to search for it in the {\tt.tex}
file), you can say \ttb{showlabeltrue} and from then on the name you used will
appear, quite small, nearby in the left margin of the printed document.

\showlabelfalse
(\label\gone/) Guess what you type to turn off this feature?

(\label/), (\label/), $\ldots$.

($\infty$) Finally, you may wish to refer to the very equation- or
item-numbers named in this {\tt.tex} file in other files. You would want to
write them as definitions into a file \keyword{filename} which you would then
\ttb{input}\ \keyword{filename} in the other file. For this to work here,
you would first say something like\par
\ttb{rememberchaptertrue} \ttb{newwrite}\ttb{chpaux}\par
\ttb{immediate}\ttb{openout}\ttb{chpaux=}\keyword{filename}\par
\noindent when you make all those other definitions in this section.
The macro \ttb{label} then takes care of the rest.

($\infty+1$) Actually, the way it is set up in this section, the name
saved is given the suffix \ttb{chapterno}, with the assumption that
\ttb{chapterno} is some word identifying this particular file. Furthermore,
the number saved is prefixed by \ttb{chapterno.}  . This means that you can
safely use names in the present file without worrying about the fact that
you might use the same name for something else in the other file. Enough
already.
{\def\bye{\vfill\eject}
\bye
\input boxes
\input errors
}
\bye
```