MATH549 Exercise Sheet 2

Deadline for submission: Monday 13th October

Some more mathematical tricks

Start a new LaTeX document called `yourname2.tex` (so I'd call it `hall2.tex`) with the `article` document class, using 11 point type and the A4 paper option. Begin a (numbered) section in the document called 'Brackets and Arrays'.

Exercise 2a

When you enclose part of a formula in brackets, you normally want those brackets to be the same height as the formula itself. LaTeX provides an easy way of ensuring this with the `\left` and `\right` commands. Here's an example:

```
\[ \left( \frac{x+y}{x-y} \right)^3.\]
```

produces

$$\left(\frac{x+y}{x-y}\right)^3.$$

The round brackets after `\left` and `\right` specify the sort of large brackets you want (there are other possibilities: square brackets, braces, modulus signs ('straight' brackets), and you can have different types at the left and the right). LaTeX calculates the height of the expression between the `\left` and the `\right`, and fits brackets of the correct size. Table 3.8

`\left`s and `\right`s can be nested, but must always come in pairs: if you only want a bracket at one end, you can specify an invisible bracket with a dot. Thus

```
\[k=\left( \frac
  {f+\left.\frac{g+h}{g-h} \right|_X}
  {f}
\right)^{-1}.\]
```

yields

$$k = \left(\frac{f + \left.\frac{g+h}{g-h}\right|_X}{f} \right)^{-1}.$$

Add a line to your file which produces the formula

$$\left(\frac{1 + \left.\frac{\partial f}{\partial y}\right|_{(0,0)}}{1 - \left.\frac{\partial f}{\partial x}\right|_{(0,0)}} \right)^2 .$$

Use `\partial` to get the partial ∂.

Exercise 2b

Large brackets often live around matrices, which are constructed using the `array` environment (environments will be the next topic covered in this sheet). Here's an example:

```
\[
\left(
\begin{array}{cc}
2 & 1 \\
1 & 0 \\
0 & 1
\end{array}
\right).
\]
```

produces

$$\left(\begin{array}{cc} 2 & 1 \\ 1 & 0 \\ 0 & 1 \end{array} \right).$$

After `\begin{array}`, there follows a pair of braces which contain as many letters as there are to be columns in the array. Each letter can be `c`, which causes the *central* points of each element in that column to be aligned, or `l` or `r`, which aligns the left or right points of the elements.

Next the elements of the array are specified, row by row, with `&` separating each element from the next, and `\\` denoting the end of the row (you don't need, indeed you mustn't have, a `\\` at the end of the last row).

By the way, there's nothing to stop you from writing the above example as

```
\[\left(\begin{array}{cc}2&1\\1&0\\0&1\end{array}\right).\]
```

if you like that sort of thing.

Mimic this example to include three copies of a 2 by 2 matrix in your document with entries $2-\lambda$, 1, 1, and $2-\lambda$. Make the elements in the first copy centre aligned, in the second copy left aligned, and in the third copy right aligned.

Exercise 2c

You're not obliged to make every `array` a matrix: here's something else you can do with them:

```
\[ |x| = \left\{
\begin{array}{rl}
x & \mbox{ if }x\ge 0\\
-x & \mbox{ otherwise.}
\end{array}
\right. \]
```

$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{otherwise.} \end{cases}$$

Your turn. Add the following formula to the end of your document:

$$\Theta(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases}$$

Environmental Change

`array` is an example of an *environment*, of which there are many in LaTeX. Each environment has a name (such as `array`), and you enter and leave the environment with `\begin{name}` and `\end{name}` respectively. Within the environment, certain symbols and commands (`&` and `\\` in the case of `array`) take on special meanings. The `\begin{name}` may also be followed by some arguments (such as `cc` in the case of `array`) which further customize the behaviour within the environment. In the next few steps you'll see how some other useful environments work. Some environments only work in text mode, others (including `array`) only in maths mode, and some move you from text mode to maths mode.

Exercise 2d

There are three environments for producing formatted lists, of which the most useful in mathematical writing is `enumerate` (I recommend that you read about the others, `itemize` and `description`, on page 39). Here's an example:

```
The following are equivalent:
\begin{enumerate}
\item $A$ is closed.
\item $X\setminus A$ is open.
\item If $(x_i)$ is a convergent sequence in $X$ with
$x_i\in A$ for all $i$, then its limit is contained in $A$.
\end{enumerate}
```

The following are equivalent:

1. A is closed.
2. $X \setminus A$ is open.
3. If (x_i) is a convergent sequence in X with $x_i \in A$ for all i, then its limit is contained in A.

Start a new (numbered) section called 'Environments', and produce:

Let $f\colon M \to M$ be a homeomorphism. Then the following are equivalent:

1. f has a fine sequence of filtrations.
2. f does not have any C^0 Ω-explosions.
3. $\Omega(f) = R(f)$.

Exercise 2e

If you want a displayed formula to be numbered, then put it in the `equation` environment:

```
\begin{equation}
\gamma_1(n)=T_1^n(x_1)-\sum_{j=1}^n T_1^{n-j}(v_1(j)).
\end{equation}
```

$$\gamma_1(n) = T_1^n(x_1) - \sum_{j=1}^{n} T_1^{n-j}(v_1(j)). \tag{1}$$

Copy this example to the end of your document. Later on you'll learn how to refer to the equation number from elsewhere in your document.

Exercise 2f

In the `verbatim` environment, LaTeX forgets that it's clever, and simply reproduces everything you write as though it were a typewriter. Thus

```
\begin{verbatim}
\[\left(\begin{array}{cc}
  2&1\\1&0\\0&1
\end{array}\right)\]
\end{verbatim}
```

gives

```
\[\left(\begin{array}{cc}
  2&1\\1&0\\0&1
\end{array}\right)\]
```

I've used `verbatim` a lot in writing these notes, and if you're an M.Sc. student then you will too when you write your Maple-LaTeX project and want to include Maple code. If the text you want to reproduce verbatim is short (for example, a single command name), then it's easier to use the `\verb` command. Look this up on page 41, and add a line to your file (using `\verb`, not the `verbatim` environment!) which produces:
I'm using `\verb`.

Exercise 2g

The `eqnarray*` environment is useful when you want to display several equations one after another, and have them nicely aligned; or when you're doing a running calculation over several lines and want all the equals signs aligned. Again, it's easier to see from an example:

```
\begin{eqnarray*}
|x_1-y_1| &<& |x_2-y_2| \\
|f_1(x)-f_1(y)| &\le& (\lambda+\epsilon)|x_2-y_2| \\
&<& (\lambda^{-1}-\epsilon)|x_2-y_2|.
\end{eqnarray*}
```

produces

$$\begin{aligned} |x_1 - y_1| &< |x_2 - y_2| \\ |f_1(x) - f_1(y)| &\le (\lambda + \epsilon)|x_2 - y_2| \\ &< (\lambda^{-1} - \epsilon)|x_2 - y_2|. \end{aligned}$$

As you can see, it works very much like `array`: there are two `&`s on each line, which surround the symbols to be aligned. `eqnarray*` is the starred version of the `eqnarray` environment, which adds an equation number to each of the lines.

Add the following to the end of your document:

$$\begin{aligned} d[\varphi(x), \varphi(x')] &= d[\theta(x, \varphi(x)), \theta(x', \varphi(x'))] \\ &\le d[\theta(x, \varphi(x)), \theta(x', \varphi(x))] + d[\theta(x', \varphi(x)), \theta(x', \varphi(x'))]. \end{aligned}$$

(Worried about how to get the x'? Just use the ' key.)

Exercise 2h

The `tabular` environment enables you to create tables. How about this? page 42

```
\begin{tabular}{|l||cccc|}
\hline
Property & 2 & 3 & 4 & 5 \\
\hline
Wecken & $\chi(M)\ge 0$ & Yes & Yes & Yes \\
totally non-Wecken & $\chi(M)<0$ & No & No & No \\
boundary-Wecken & $\chi(M)\ge0$ & $\chi(C)\ge0$ & Yes & Yes\\
totally non-boundary-Wecken & $M=K\setminus\{\mbox{discs}\}$ &
   $M$ orientable & No & No \\
\hline
\end{tabular}
```

Property	2	3	4	5
Wecken	$\chi(M) \geq 0$	Yes	Yes	Yes
totally non-Wecken	$\chi(M) < 0$	No	No	No
boundary-Wecken	$\chi(M) \geq 0$	$\chi(C) \geq 0$	Yes	Yes
totally non-boundary-Wecken	$M = K \setminus \{\text{discs}\}$	M orientable	No	No

Once again, it works rather like `array`: you can include the character `|` in the argument of `tabular` to specify where vertical lines should be placed, and put horizontal lines between rows using the `\hline` command. There are plenty of other options, but I use them so seldom that I have to look them up each time. Note that `tabular` works in text mode: if you want mathematics in your tables, you need to enclose each mathematical entry in $ signs.

For your example, try to produce the following table:

P	c_P	π_P	Type	$\rho(P)$	$r(P)$
s_5^1	10110	(13425)	fo	$\{2/5\}$	$1/2$
s_5^2	10010	(12435)	pA	$[1/3, 1/2]$	$1/3$
s_5^3	10001	(12345)	fo	$\{1/5\}$	$1/2$

How do you get it centred on the page? Put it in a `center` environment, of course. Make sure the set brackets { and } have come out in your document. If they didn't, why not?

There are a couple more environments to learn about, namely `thebibliography` and `figure`, but they'll come later.

Making all things new

Not enough commands and environments for you? Why not define some of your own? This can get pretty complicated, but we'll stick to the rudiments.

Exercise 2i

At the simplest level, you can define a new command just to cut down on typing. I often write documents with lots of ϵ's in them, and I get fed up of typing `\epsilon` all the time. So I define a new command like so: page 106

```
\newcommand{\eps}{\epsilon}
```

and I can just use `\eps` as an abbreviation for `\epsilon`. That is, whenever LaTeX sees `\eps`, it treats it exactly as though `\epsilon` were there instead.

The `\newcommand` command takes two arguments: the first is the name of the new command you're defining (here `\eps`). Whenever LaTeX sees this new command, it simply replaces it with the second argument of `\newcommand`.

Here's another convenient one:

```
\newcommand{\I}{^{-1}}
```

Whenever LaTeX sees `\I`, it treats it exactly as though `^{-1}` were there instead, so I can type `$x\I$` instead of `$x^{-1}$` to produce x^{-1}. Again, I find this saves time.

You can put the `\newcommand` anywhere in your file before the first time the new command is used, but it makes sense to keep all your new command definitions together, either in the preamble or in a separate file. Start a new section in your file called 'All things new', and define new commands `\hm` and `\Gp` in your preamble (i.e. the part of the file before `\begin{document}`) so that adding the following line

```
Let $f\Gp$ be a \hm.
```

at the end of your file produces
Let $f: G \to G$ be a homomorphism.

(If you have difficulty with this, write down what you'd need to type to produce 'Let $f: G \to G$ be a homomorphism' without any `\newcommand`s. Then define `\Gp` to be replaced by exactly those symbols which come between `$f` and `$` in what you've written down.)

Exercise 2j

LaTeX commands can also take arguments. Suppose you're writing a document which contains lots of vectors in the form $(x_1, \ldots, x_n)$, $(y_1, \ldots, y_n)$, and $(z_1, \ldots, z_n)$. You'd find it convenient to be able to type `$\Vect{w}$` to produce $(w_1, \ldots, w_n)$. You can. Just enter the following command definition:

```
\newcommand{\Vect}[1]{(#1_1,\ldots,#1_n)}
```

The `[1]` tells LaTeX that the command you're defining takes one argument. The `#1` in the definition is simply replaced by whatever argument the command is given. Thus if you type `\Vect{q}` anywhere in your document, LaTeX replaces it exactly with `(q_1,\ldots,q_n)`.

More arguments can be added in the same way. For instance, suppose now that you find you've started to use vectors with different numbers of elements. No problem.

```
\newcommand{\VECT}[2]{(#1_1,\ldots,#1_{#2})}
```

Now `$\VECT{x}{n}$` produces $(x_1, \ldots, x_n)$, and `$\VECT{\xi}{m}$` gives $(\xi_1, \ldots, \xi_m)$.

A couple of quick points. First, LaTeX has a lot of commands, and you may find that your favoured new command names are already taken. If they are then LaTeX complains. One way to avoid this most of the time is to include at least one capital letter in each command name. Remember, too, that command names must consist exclusively of letters.

Second, it doesn't harm to put lots of extra braces around everything in command definitions. For example, suppose you defined `\VECT` by

```
\newcommand{\VECT}[2]{(#1_1,\ldots,#1_#2)}
```

(i.e. missing out the extra braces around `#2`). Now, when you type `$\VECT{x}{mn}$`, LaTeX sees `$(x_1,\ldots,x_mn)$`, which comes out as $(x_1, \ldots, x_m n)$: not what you intended.

Try defining a new command `\Charpoly` so that when you add the line

```
\[\Charpoly{2}{-1}{3}{1}{\lambda}=\lambda^2-3\lambda+5.\]
```

it produces

$$\begin{vmatrix} 2-\lambda & -1 \\ 3 & 1-\lambda \end{vmatrix} = \lambda^2 - 3\lambda + 5.$$

Notice that your command doesn't have to *calculate* the characteristic polynomial: you're *telling* LaTeX that the result is $\lambda^2 - 3\lambda + 5$. (There's a hint for this exercise on the module webpage: have a look at it if you're having problems.)

Exercise 2k

Defining new commands takes a little thought, and once you've made up some good ones it makes sense to keep them in a separate file, and use them in all of your documents. You can do this using the `\input` command. Cut all your command definitions out of the preamble, paste them into a new file called `yournamedefs.tex`, and replace them in your file with the line

```
\input{yournamedefs}
```

(Thus if I were doing this, I'd put the definitions in a file called `halldefs.tex`, and replace them with the line `\input{halldefs}`.) The effect of the `input` command is to read in the specified file and treat it exactly as though its contents were there instead of the `\input` command. Check that the new arrangement still works.

Exercise 2l

You can define new environments too. In this case, you have to tell LaTeX what you want it to do both at the beginning and the end of the environment, so the `\newenvironment` command takes three arguments: the environment name, what it has to do at the start of the environment, and what it has to do at the end. Here's one I use a lot:

```
\newenvironment{proof}
{\par\noindent\emph{Proof.\ }}
{\hfill\rule{1.2mm}{2.2mm}\vspace{1.4mm}\par\noindent}
```

It doesn't harm to put the different arguments of a command on separate lines for clarity. The first argument gives the name of the environment (`proof`), the second says that when I type `\begin{proof}` I want *Proof.* to be written at the start of a new non-indented paragraph, and the third that when I type `\end{proof}` I want a little black box at the end of the line (don't worry about how this one works: I don't remember myself, and it was written years ago in the days when LaTeX was more primitive than it is now, so could probably be done more easily. But that's the point — once you have a definition which works, you never need to think about it again). So when I type

```
\begin{proof}
Multiply all the numbers together, integrate from $0$ to $\infty$, and
the result is obvious.
\end{proof}
```

I get

Proof. Multiply all the numbers together, integrate from 0 to ∞, and the result is obvious. ∎

Now you try it. Define a new environment `defn` in `yournamedefs.tex` so that you can type

```
\begin{defn}
An \emph{irrational number} is a real number which isn't rational.
\end{defn}
```

and get

Definition
An *irrational number* is a real number which isn't rational.

This may not seem a great time-saver, but it has one great benefit. Once you've created an environment like this, you never need to think again about how to format your definitions: you simply declare them to be definitions, and they all come out the same way. If you decide you want to change that way, you only have to make the change once in `yournamedefs.tex`, and all your definitions automatically come into line.

Exercise 2m

You may never need to define new commands and environments, but if you're working in Pure Mathematics you will almost certainly want to use the `\newtheorem` command. This creates a special type of environment which is suitable for displaying and numbering 'theorem-like' statements.

Put the following in `yournamedefs.tex`:

```
\newtheorem{thm}{Theorem}
```

and then add to the end of your file:

```
\begin{thm}
The orbit bijects with the cosets of the stabilizer.
\end{thm}
\begin{thm}[Cayley-Hamilton]
Every matrix satisfies its own characteristic equation.
\end{thm}
```

Process the file and study the result. Notice that the theorems are numbered consecutively and are displayed in an appropriate font.

The first argument of `\newtheorem` gives the name of the new environment, and the second gives the text which will be displayed in theorem statements. Notice that when you start a 'theorem-like' environment, you can give an optional argument (`Cayley-Hamilton` in the case above) which typically gives the name of the theorem or of its discoverer.

`\newtheorem` can also take two optional arguments. To understand the first, add the following line to `yournamedefs.tex` (*after* the `\newtheorem` command which is already there):

```
\newtheorem{lem}[thm]{Lemma}
```

and then add to your file

```
\begin{lem}
Homotopy is a congruence on the category Top.
\end{lem}
```

between the two theorems you already have. Process the file and study the result. The optional `[thm]` argument to `\newtheorem` specifies that the numbering of theorems and lemmas should use the same counter. To hammer the point home, add `\newtheorem{aside}{Aside}` to `yournamedefs.tex`, and add an aside between the first theorem and the lemma, stating 'I didn't write this, you know.' Process and study.

To understand the second optional argument of `\newtheorem`, change your first `\newtheorem` command to

```
\newtheorem{thm}{Theorem}[section]
```

Process the file, and notice that the theorem and lemma numbers are now based on sections. Cut the lemma and paste it at the end of the first section of your document, process, and study.

Labels and Citations

Once you have numbered theorems in your document, you'll probably want to refer to them, along the lines of 'the result follows by Lemma 1 and Theorem 7'. A difficulty arises when you decide that you really need a new lemma before Theorem 7: Theorem 7 is suddenly relabelled Theorem 8, and you have to search though your file for all references to Theorem 7 and change them. LaTeX provides a good way of getting around this problem.

Exercise 2n

Add a line to your earlier theorem statement so that it reads:

```
\begin{thm}
\label{thm:orbstab}
The orbit bijects with the cosets of the stabilizer.
\end{thm}
```

Start a new section in your document called 'Labels', and add the line

```
Theorem~\ref{thm:orbstab} is very useful.
```

You'll learn what the *tie* `~` is doing there later, but you should always use a tie rather than a space in situations like this, and I want you to get into good habits from the start.

Process the file and study the result. Oops — that **??** isn't what you wanted. Process the file a second time and it will come right.

What's happening? When LaTeX reaches the `\label` command, it adds a line to an *auxiliary* file called `yourname2.aux` which associates the label `thm:orbstab` with the theorem number (3.1) (and also with the page number on which the theorem appears). Every time you run LaTeX on `yourname2.tex`, it starts off by reading `yourname2.aux` to gather the information about labels. The first time you ran LaTeX, the information about `thm:orbstab` wasn't in the auxiliary file, so it couldn't fill in the correct theorem number. Likewise, if you now put another theorem before Theorem 3.1 (thus changing it to Theorem 3.2), it would take two runs of LaTeX to get the reference right.

Give the label `lem:homtop` to the lemma in the first section of your document, and add the following at the end of your file:

```
Lemma~\ref{lem:homtop} (on page~\pageref{lem:homtop}) is silly.
```

Process (twice) and study. There's nothing magic about the labels `thm:orbstab` and `lem:homtop`: you could equally well have used the labels `squirrel` and `rabbit`. When you have a lot of labels in your document, though, it makes sense to give them names which relate to what they're labelling, so you can remember them more easily.

Exercise 2o

It isn't just theorems that you can refer to like this. Remember that numbered equation in the section on environments? Insert `\label{eqn:gam}` after `\begin{equation}`, and add a line to your file to produce
There is a mistake in equation (1).

Put another `\label` statement after `\section{Environments}`, and add a line which gives
Section 2 is my favourite so far.

The rule is that, if a `\label` appears in an environment which has an associated number, then that is the number the label refers to (there's an important exception to this for the `figure` environment, which you'll meet in sheet 3). Otherwise, it refers to the number of chapter, section, subsection etc. corresponding to the most recent sectioning command. The page number associated to a label is always the page number of the exact place in the text where the label occurs.

Exercise 2p

Citations (references to entries in a bibliography) work similarly. First, let's create a short bibliography. Put several blank lines just before `\end{document}` (everything you enter after this step should come before the bibliography), and enter the following (*before* `\end{document}`):

```
\begin{thebibliography}{8}
\bibitem{BelMey} Bell, H. and Meyer, K.
``Limit periodic functions, adding machines and solenoids.''
\textsl{J. of Dynamics and Differential Equations}
\textbf{7} (1995), 409\,--\,415.
\bibitem{BestHan} Bestvina, M. and Handel, M.
``Train tracks for surface homeomorphisms.''
\textsl{Topology} \textbf{34} (1995), 109\,--\,140.
\end{thebibliography}
```

Then add the following line to the end of your file (but *before* the bibliography):

```
The interested reader should consult~\cite{BelMey} or~\cite{BestHan}
for further details.
```

Process (twice) and study. The `\,` is a subtlety which will be touched on later.

The argument `8` to `thebibliography` is required to be some text which is at least as wide as the widest label which is going to appear in the bibliography. In this case, the labels are **1** and **2**, so 8 is wide enough: if you had ten or more references, you should use 88 instead. Some people prefer more informative labels than consecutive numbers: LaTeX enables you to do this with an optional argument to the `\bibitem` command. Change the bibliography in the following ways:

```
\begin{thebibliography}{BM}
\bibitem[BM]{BelMey} Bell ...
...
\bibitem[BH]{BestHan} Bestvina ...
```

Process the file twice, and see what has changed. Notice the argument `BM` to `thebibliography`: this is probably the wider of the two labels BM and BH.

By the way, typing bibliographies in this way is very tedious: in Sheet 3 there'll be brief details on a much easier approach.

Exercise 2q

Add a title and your name to the document. Submit the assignment to me by email, attaching *both* the files `yourname2.tex` and `yournamedefs.tex`.

Aesthetics

That's more or less it for this week. This final section contains a few miscellaneous guidelines for improving the appearance of your documents. They may be miscellaneous, but they're important! I'll assume in next week's sheet that you've read this section. I hope that by this stage you're finding LaTeX isn't too hard. Because of this, I pay more attention than you might expect to fussy little details, like those in this section, when assessing the LaTeX component of M.Sc. Maple-LaTeX projects.

Changing the default layout

There are an enormous number of commands which enable you to change the way in which LaTeX lays out the document. I deliberately haven't touched on them because, unless a career as a designer beckons, you'd be certain to make your documents look worse rather than better. There are, however, a couple of ways in which I break my rule of not messing around with the workings.

Some people consider the default text width in LaTeX to be rather narrow (but see page 116 for another opinion). To make it a bit wider, you could put the following commands into the preamble:

```
\addtolength{\hoffset}{-0.7cm}
\addtolength{\textwidth}{1.4cm}
```

This increases the width of your text by 1.4cm. If you only included the second line, the left hand margin of your text would be unchanged, so it would be unbalanced on the page: the first line moves the left margin 0.7cm to the left to compensate for the wider text. There's a diagram on page 117 which illustrates what `\hoffset` and `\textwidth` are, as well as a lot of other measurements which you can tweak. LaTeX recognizes a whole host of units when you're specifying lengths: there's a partial list of them on page 115.

Secondly, you might like the lines to be a little more widely spaced than they are by default. The command which does this is called `\openup`, and the units in which opening up is measured are called `\jot`s. Depending on the document, you may want to open up by between 1 and 2 jots: thus you might have something like

```
\openup 1.2\jot
```

in your preamble.

Ties and mboxes

You've already seen the tie `~` in action: it is exactly the same as a space, except that you are guaranteed that the two words on either side of it will lie on the same line. 'Theorem 3.1' doesn't look good if 'Theorem' comes at the end of one line and '3.1' at the beginning of the next: so you put a tie between them: `Theorem~3.1`. Similarly, you should write `Dr.~Hall` and `J.~G.~Witherspoon`.

In the example above the 'Theorem' in 'Theorem 3.1' could still end up being hyphenated between two lines. When there is some piece of text which must under all circumstances appear on a single line, you should put it in an `\mbox`: for example, `My phone number is \mbox{0151 794 4065}`. Naturally, if the text in an `\mbox` is very long, there's no guarantee it will look good. For instance,

```
\mbox{This text will all appear on the same line, but that doesn't
mean you'll be able to read all of it at the same time}
```

gives

This text will all appear on the same line, but that doesn't mean you'll be able to read all of it at th

Dashes

LaTeX provides three sizes of dashes, which you get by typing `-`, `--`, and `---`. The shortest is the hyphen: `middle-aged` produces middle-aged. The second one is used to typeset page ranges: `pages~23\,--\,32` produces pages 23 – 32. You should usually include the two `\,` in such ranges: they insert a small amount of extra space (see later). Finally, the largest dash is the punctuation dash:

```
Don't ask me what a quad is --- I don't know.
```

produces: Don't ask me what a quad is — I don't know.

Dots

You should never try to produce an ellipsis (...) by typing `...` (well, you could try it once just to see how bad it looks) — the command for doing this is `\ldots`. In maths mode, there's a second type of ellipsis called `\cdots`, which produces the dots at the centre of the line rather than the bottom. You should use `\ldots` between commas and when symbols are juxtaposed (thus `$a\ldots z$` and `$x_1,\ldots,x_n$` should be used to obtain $a \ldots z$ and $x_1, \ldots, x_n$); and you should use `\cdots` between other symbols such as $+$, $\times$, $=$, $\leq$, and $\oplus$ (thus `$x_1+\cdots+x_n$` and `$I_0\le\cdots\le I_r$` for $x_1 + \cdots + x_n$ and $I_0 \leq \cdots \leq I_r$).

Spacing in formulae

Sometimes the symbols in a formula appear too bunched up or too widely spaced: if you notice this, you can correct it by using `\!`, a thin negative space, which pulls things closer together, or `\,`, a thin positive space, which pushes them apart. (There are also `\>` and `\;`, which are slightly wider positive spaces, but you rarely need them). Thus `$\sqrt{2}\,x$`, producing $\sqrt{2}\,x$, looks a little better than $\sqrt{2}x$, and `$x^2\!/2$`, producing $x^2/2$, is better than $x^2/2$. Well, maybe.

On the whole, it's difficult to predict where these little spaces will be needed: you should add them in at the end if any of your formulae don't look quite right. One area of consistency, however, is that formulae involving calculus almost always look better when you put a `\,` before each d: thus `$\int f(x)\,dx$` for $\int f(x)\,dx$ and `$y\,dx-x\,dy$` for $y\,dx - x\,dy$. In these cases, it's probably easier to put the space in when you type the formula, and take it out afterwards if it doesn't look right.

A separate issue is that of spacing between different parts of a displayed formula, as in

$$F_n = F_{n-1} + F_{n-2}, \qquad n \geq 2.$$

The space between the formula and the side-condition isn't inserted automatically. The commands to use for such spaces are `\quad` and `\qquad`, which insert one and two quads of space respectively (don't ask me what a quad is — I don't know). The above example was typeset as `\[F_n=F_{n-1}+F_{n-2},\qquad n\ge 2.\]`