

# COMP10120 Lab Session 3

## Report writing and Digital Typography

### Contents

---

<b>3.1</b>	<b>Introduction . . . . .</b>	<b>8</b>
<b>3.2</b>	<b>Getting started . . . . .</b>	<b>9</b>
	Exercise 1: A closer look . . . . .	9
	Exercise 2: A larger example . . . . .	12
<b>3.3</b>	<b>Common formatting features . . . . .</b>	<b>22</b>
	3.3.1 Enumerated and bulleted lists . . . . .	22
	3.3.2 Changing the appearance of text . . . . .	23
	3.3.3 Cross referencing . . . . .	23
	3.3.4 Figures and pictures . . . . .	25
	Exercise 3: Including images . . . . .	26
	Exercise 4: A reflection on your studies	27

# COMP10120 Report writing and Digital Typography

---

<b>3.4</b>	<b>Typesetting Mathematics</b>	<b>28</b>
3.4.1	A longer piece of mathematics	32
	Exercise 5: Formatting mathematics	34
<b>3.5</b>	<b>Some optional extras</b>	<b>35</b>
	Exercise 6: Exploring more $\text{\LaTeX}$ features (optional, but easy)	35
	Exercise 7: A neat directory listing (optional, more challenging)	36
	Exercise 8: A cleaning rota (optional, even more challenging)	37
3.5.1	Marking Scheme	37
<b>3.6</b>	<b>Next steps</b>	<b>38</b>
<b>3.7</b>	<b>Acknowledgements</b>	<b>39</b>

---

These notes are available online at

[studentnet.cs.manchester.ac.uk/ugt/COMP10120/labscripts/10](http://studentnet.cs.manchester.ac.uk/ugt/COMP10120/labscripts/10)

## 3.1 Introduction

$\text{\LaTeX}$  is a document preparation system that is fundamentally different to anything that you are likely to have seen before. It's used worldwide by publishers, academics and scientists and it happens to be very good for writing CVs too.

## 3.2 Getting started

Please complete all the exercises in this lab in your `COMP10120/ex3` directory. When you have completed as much as you are going to do, you should run `submit` and (when next in the School) `labprint`.

### Exercise 1: A closer look

Fire up your favourite editor, enter the following text (it's important that you enter it exactly as it is below for now), and then save it as a file called `fire-and-ice.tex` (in case you're wondering, it's the first line from a short poem called 'Fire and Ice' by American poet Robert Frost (R. Frost, 1924); the whole poem is on the Poetry Foundation website if you're interested.)

```
\documentclass[a4paper]{article}
\begin{document}
Some say the world will end in fire,
\end{document}
```

Next, run the command `pdflatex fire-and-ice.tex` (for now it's sufficient to say that `pdflatex` invokes  $\text{\LaTeX}$  to create a PDF file as the output; the full story is a bit more complicated.)

## COMP10120 Report writing and Digital Typography

---

$\text{\LaTeX}$  will process your file, and print out a surprisingly large amount of text into the shell window as it does so. If all is well, the last two lines printed out will be something like:

```
Output written on fire-and-ice.pdf (1 page, 11853
Transcript written on fire-and-ice.log.
```

after which  $\text{\LaTeX}$  will return you to the command prompt. If you've made a mistake typing in the file,  $\text{\LaTeX}$  will stop part way through processing your work, and display an error (possibly, quite an obscure one) and show a '?' prompt. The best thing to do at this stage is to press `<ctrl>d`, or `x`, to tell  $\text{\LaTeX}$  to give up trying to process your file. Then fix the mistake, and try again.

If you now list the directory contents, you should see that a file `fire-and-ice.pdf` has been created (along with two other files, `fire-and-ice.aux` and `fire-and-ice.log` both of which we will ignore for now). Use one of the PDF readers (such as `evince`) to look at the contents of `fire-and-ice.pdf`. Admittedly, it's not the most exciting result; if everything has worked properly so far, you should see a single-page document with the words 'Some say the world will end in fire,' a little way down from the top of the page<sup>1</sup>.



---

<sup>1</sup>If you have been carried away with the sheer excitement and joy of learning to use  $\text{\LaTeX}$  and improvised the text rather than typing the first line of 'Fire and

But now look more closely. A lot more closely... really zoom in. See anything interesting?

Even for this very short document, L<sup>A</sup>T<sub>E</sub>X has taken a fairly sophisticated typographical decision on your behalf and **ligated** (which is typography-speak for ‘joined’) the ‘f’ and the ‘i’ in the word ‘fire’; the spacing between the characters has been subtly altered so that the dot above the ‘i’ and the blob on the end of the curvy bit (the ‘arc of the stem’, if we’re being formal) on the top of the ‘f’ join together to form a single shape called a **ligature**. Why? Because having two blobs side by side simply looks a bit clumsy as you can see in Figure 3.1. To make this decision, L<sup>A</sup>T<sub>E</sub>X has had to know a lot about the fonts being used (not all ‘f’s in all fonts have a blobby end; not all ‘i’s in all fonts have a round dot that would merge nicely with the blobby bit on the end of an ‘f’, and so on). There’s a reasonable chance that you’ve never heard of ligated characters before—it is, after all, a fairly specialist thing. And there are hundreds of other obscure but important ‘rules’ of typography that go to make professionally typeset documents look good<sup>2</sup>. Individually, they might not be

---

Ice’ as requested, please go back and change it now: it really is important that you use exactly that text. Really, it is. That’s better. Thanks.

<sup>2</sup>Look up ‘kerning’, ‘combining characters’ and ‘serif’ on Wikipedia for starters. And if you’re really keen, Donald Knuth’s fascinating and compendious book ‘Digital Typography’ (D. Knuth, 1999) has an entire chapter dedicated to the joys of the letter ‘S’. Probably best not to bring this subject up at the

obvious or hugely important, but collectively and subliminally they make the difference between something that looks just-about-acceptable (like most things written using wordprocessors) and things that stand out as looking really professional (you might want to remember this when it comes to putting your CV together.)

fire fire

(a) (b)

**Figure 3.1**

The word ‘fire’ typeset by (a) Microsoft Word 2011 without ligated characters, and (b) by  $\text{\LaTeX}$  showing the ligation of the characters ‘f’ and ‘i’.

But that’s enough about the typography of the word ‘fire’ for now. Let’s put together something more substantial.

### Exercise 2: A larger example

Create yourself a new file, called `sections.tex` by copying `fire-and-ice.tex`. In `sections.tex`, replace the line of poetry by copying and pasting the text from Figure 3.2 (which contains the first three paragraphs of Dickens’ novel, ‘Bleak House’ (C. Dickens, 1852)).

Edit the text to make sure that there is a blank line between each of the three paragraphs (it’s not enough

---

pub though, or at parties. Unless they are very specialist parties.

*London. Michaelmas term lately over, and the Lord Chancellor sitting in Lincoln's Inn Hall. Implacable November weather. As much mud in the streets as if the waters had but newly retired from the face of the earth, and it would not be wonderful to meet a Megalosaurus, forty feet long or so, waddling like an elephantine lizard up Holborn Hill. Smoke lowering down from chimney-pots, making a soft black drizzle, with flakes of soot in it as big as full-grown snowflakes—gone into mourning, one might imagine, for the death of the sun. Dogs, undistinguishable in mire. Horses, scarcely better; splashed to their very blinkers. Foot passengers, jostling one another's umbrellas in a general infection of ill temper, and losing their foot-hold at street-corners, where tens of thousands of other foot passengers have been slipping and sliding since the day broke (if this day ever broke), adding new deposits to the crust upon crust of mud, sticking at those points tenaciously to the pavement, and accumulating at compound interest.*

*Fog everywhere. Fog up the river, where it flows among green aits and meadows; fog down the river, where it rolls defiled among the tiers of shipping and the waterside pollutions of a great (and dirty) city. Fog on the Essex marshes, fog on the Kentish heights. Fog creeping into the cabooses of collier-brigs; fog lying out on the yards and hovering in the rigging of great ships; fog drooping on the gunwales of barges and small boats. Fog in the eyes and throats of*

that each paragraph starts on a new line of its own, there has to be an empty line in between them). That's how paragraphs are distinguished in  $\text{\LaTeX}$  (in fact, it doesn't matter how many blank lines you put as long as there is at least one... for the most part  $\text{\LaTeX}$  ignores spare whitespace). After the 'Bleak House' text, type a paragraph of gibberish by randomly hitting keys on the keyboard, and putting spaces in to make 'words' in the text file. You should end up with something like the following, but about three times as long (don't cut and paste this from here... it's important that you actually create some unique gibberish yourself!):

```
Oisjdf oqweqwe oi soijs hbweo kbsd oijsdf  
oijqwknpiouh iusbdfspb sifuhygwqeb usgweijf  
blimqwoq oieuerwefwui aokqjw uioshiufds  
qiqks odsubfi psdiweneq.
```

Be careful at this stage to only include letters (uppercase and lowercase), commas, full-stops and perhaps numbers in your gibberish text: if you use any other symbols, you may upset  $\text{\LaTeX}$ .

Next go to the website [www.lipsum.com](http://www.lipsum.com) and follow the instructions to generate yourself three paragraphs of what's called **Lorem Ipsum** text. Copy and paste that text into your  $\text{\LaTeX}$  document too (don't worry about what it means at this stage), again making sure that you have a blank line between the paragraphs.



## COMP10120 Report writing and Digital Typography

---

Run `pdflatex` and look at the resulting `sections.pdf`. Again, at first glance, there's nothing hugely exciting going on here: the text from your `.tex` file has been assembled into a (probably) two page PDF document. But in fact quite a few things have happened:

- The blank lines that separate the paragraphs in your source file are gone, and instead  $\text{\LaTeX}$  has indented the first line of each paragraph. There's nothing profound here about the decision to indent paragraphs rather than (say) to leave a gap between them (so called **block paragraphs**); it's just  $\text{\LaTeX}$ 's default paragraph style, and can be changed easily enough.<sup>3</sup>
- There are page numbers at the bottom of each page (except page 1). This is a very sensible default, allowing you in meetings to ask people to 'turn to page 4' and so on (it's a bit odd, and a source of some annoyance, that most wordprocessors don't do this by default too). But you can easily switch page numbers off, change their style or move them elsewhere if you need to.
- The text is **fully justified** so that it's flush on both the left and right hand edges.

---

<sup>3</sup>Though apparently, there is some evidence in M. A. Tinker, 1963 that indented paragraphs increase readability

- Certain words that fall at the end of lines have been hyphenated, even if they weren't originally split by hyphens in the source text.

These last two points about justification and hyphenation might seem utterly trivial, but getting these factors right turns out to be an important part of making readable documents; readability is not just about which font you choose, but about how the characters and spaces are distributed on each line. It's something that you will have seen happen so often in professionally typeset materials that you'll have taken it for granted, and you may not have spotted that it doesn't happen in quite the same way when you use a wordprocessor. The principle behind justifying text seems easy enough at first glance: you simply have to take any 'left over' space that would be at the end of a line, and redistribute it between the words in some way to 'space them out' a bit more so that they take up all the available horizontal room. But it turns out to be much harder than it seems, and all the simple ways you might think of for doing this on a per-line basis give results that are visually quite unsatisfactory, with nasty patterns made up of neighbouring whitespace appearing in the layout and ruining everything. The added complication here is that to do the optimum job of shuffling the words round on their lines, you really need the flex-

ibility of being able to hyphenate words to give you extra space on lines that would otherwise be tightly packed with characters. But if you hyphenate words badly, by breaking them in inappropriate places, or do it too often then that makes the text hard to read too. It's all deeply inconvenient and inter-related. The best solution to this aesthetic conundrum so far was developed by Knuth and Plass, 1981 and later improved by Frank Liang during his PhD research (F. M. Liang, 1983), and it's a technique that tries to optimise the layout of whole paragraphs rather than individual lines. The algorithm takes into account all manner of different things: language-specific character patterns, the number of consecutive lines that end with hyphens, the word 'tightness' on each line... and many others. The details don't really matter here, but the really important insight is that you get better results by working on a bigger chunk of the document (in this case a paragraph at a time) rather than trying to independently optimise lots of small bits (say, lines) and hoping that the result of joining them all together turns out nicely. This idea of optimising things **globally** versus **locally** is something you'll see many times throughout your Computer Science degree; watch out for it in all the algorithms courses.

This highlights a fundamental difference between the WYSIWYG paradigm and the way that documents are produced using  $\text{\LaTeX}$ . As we've seen, small changes

down at the level of individual characters can have fairly large knock-on consequences for the optimal layout of a document: changing one letter for another affects the length of a line, which in turn affects the hyphenation options, which then affects the length of a paragraph, which means that the best place for a figure that was on page 99 might now be on page 100, but that means that all references to 'page 99' now need a bit more room to account for the extra digit which means that the some line lengths have changed which means. . . well, you get the idea. It's more complicated than it first seems.  $\text{\LaTeX}$  gets to see the *whole of your document at once*, and can therefore make decisions about how best to hyphenate lines, position figures or whatever else it likes *before* it shows you the results; so it can make solutions that are optimal for the whole document, as well as looking at details such as ligated characters. Wordprocessors can't afford to do this since it would be deeply distracting if every time you typed a character, the whole layout of the document flapped around in front of your eyes. Wordprocessors can only sensibly make little optimisations at a local (probably per-line level). It's not that wordprocessors are rubbish, or that the people that programmed them were lazy: it's a fundamentally different way of working, and knowing the pros and cons of the different approaches will help you pick the right tool for the job.

You may have noticed that actual text in your document (and in fact, this one) takes up a (perhaps) surprisingly narrow horizontal region of the page. This last ‘feature’ might seem odd at first, but it illustrates another important typographical point that you may never have thought of before: reading long lines of text is hard, because your eyes find it hard to move in a completely straight line as they scan from left to right. So the longer the line of words, the more you have to work to keep your eyes from accidentally drifting up or down onto a neighbouring line (perhaps you’ve experienced the phenomenon of accidentally re-reading the same line over and over as your eyes get tired). It may never have occurred to you, but most books have relatively short lines of text; and those that use large pages often split text into several columns so that individual lines are kept relatively short. The widely trusted book ‘The Elements of Typographic Style’ (R. Bringhurst, 1992) makes the following assertion:

*‘Anything from 45 to 75 characters is widely-regarded as a satisfactory length of line for a single-column page set in a serifed text face in a text size. The 66-character line (counting both letters and spaces) is widely regarded as ideal.’*

L<sup>A</sup>T<sub>E</sub>X knows about this issue, so made a sensible default decision for you. You may have noticed that there

is a subtle difference in appearance of this chapter from previous ones: the lines are shorter and paragraphs are indented. We wanted to use the default L<sup>A</sup>T<sub>E</sub>X layout for this chapter, but changed it for earlier chapters in an effort to save paper.

There's one last unusual thing to note: your paragraph of gibberish text in between the real words from Bleak House and the **Lorem Ipsum** almost certainly stands out and looks really quite odd (it might even have confused L<sup>A</sup>T<sub>E</sub>X's hyphenation rules enough to have ended up with one or more lines sticking out proud of the right hand margin). Let your eyes relax so that your vision blurs for a moment: you can still probably spot that the gibberish paragraph looks out of place. It's quite likely that you even found creating the gibberish unexpectedly hard. The interesting thing here is that as humans, we spend so much time reading and writing 'proper' natural language that we become attuned to the patterns of letters words and the underlying 'rhythm' of our written language that its very hard to artificially reproduce something that looks plausible; the frequency of the letters you chose by randomly hitting keys, and the length and patterns of the words you put in place most likely don't match the natural patterns of real languages, so they just look plain wrong. Curiously the **Lorem Ipsum** text that you got from the website probably doesn't look so bad, in spite of it not

being made up of real words. But that's not surprising since **Lorem Ipsum** is specifically created to be used as a placeholder for real text when typesetting. While the words are meaningless Latin-like things, their length, and the distribution of characters and word lengths are chosen to mimic patterns in real language.

Next let's give our document some structure by dividing it up into sections. On the line before Bleak House starts, add the command

```
\section{Bleak House}
```

making sure it appears on a line of its own. Then just before your paragraph of gibberish, put a similar instruction (again, on its own line), and then create a final section before the Lorem Ipsum. Rerun `pdflatex` and look at the results. You should see the section headings appear in the appropriate places in the PDF, automatically having been given section numbers. Experiment by adding in a couple of sub-sections towards the end of the document using the `\subsection` command, which like the `\section` command above takes as a parameter some text contained in curly brackets.

Now, at the top of your document after the `\begin{document}` line and before the command defining the first section, add the following two commands, each on their own line:

```
\tableofcontents  
\newpage
```

Re-run `pdflatex` *twice*, and check that the effect is what you'd expect. You need to process the source twice because on the first run,  $\text{\LaTeX}$  gathers and stores information about what to put in the table of contents, and only creates it on the second run.

### 3.3 Common formatting features

Before we give you some more exercises to do, here is a brief introduction to some commonly used features of  $\text{\LaTeX}$ .

#### 3.3.1 Enumerated and bulleted lists

Creating bulleted or numbered lists in  $\text{\LaTeX}$  is straightforward, and is done like this:

```
\begin{enumerate}  
\item My first item  
\item My second item  
\item The last thing on my list  
\end{enumerate}
```



You must make sure that you both `begin` and `end` your list, and that each item is terminated by a newline. Changing `enumerate` to `itemize` gives you a bulleted rather than numbered list (note the American spelling of ‘itemize’).

### 3.3.2 Changing the appearance of text

Text can be typeset in **bold** font using the `\textbf`, in *italics* using `\textit` and in a fixed-width font using `\texttt` as in the example below:

Some `\texttt{say}` the `\textbf{world}` will end in  
which is typeset as

Some say the **world** will end in *fire*,

### 3.3.3 Cross referencing

L<sup>A</sup>T<sub>E</sub>X allows you to cross-reference almost anything in the document, which includes sections, sub-sections and figures. To use the cross-referencing feature you simply insert the command

```
\label{mymarker}
```

at the point in the document you want to refer to, and then use the command

## COMP10120 Report writing and Digital Typography

```
\ref{mymarker}
```

when you want to use the reference. Obviously you replace the text `mymarker` with something more meaningful. An important tip here is to call the marker something that refers to the content of that part of the document, and to avoid the temptation to use numbers (so don't call your marker `section1`, instead call it `introduction`; if you reorder your document, it's still likely to be the introduction, but it may no longer be Section 1). For example:

```
\section{Reflection on Welcome Week}
```

```
\label{welcomeweek}
```

```
I spent most of welcome week getting to know my w  
around campus. I should have made more effort to  
societies I think; I must remember to get to the  
Students' Union and take a look around there.
```

```
\subsection{Things I enjoyed}
```

```
Everything so far has been wonderful, with the ca  
mentioned in Section \ref{welcomeweek}.
```

```
\subsection{Things I didn't enjoy}
```

```
I've discovered that I don't like soggy chips.
```

The really useful thing about cross-referencing in  $\text{\LaTeX}$  is that you'll get warnings when you compile your doc-



**Figure 3.3**

Some men, possibly Dutch, looking at cheeses.

ument if cross-references are undefined or used incorrectly. This means you don't have to manually check every reference in your document to make sure it's still right. You do, however, have to run  $\text{\LaTeX}$  twice, for similar reasons to those given above.

### 3.3.4 Figures and pictures

Including figures and pictures in your document is easy. You do it like this:

```
\begin{figure}
\centerline{\includegraphics[width=10cm]{images/m
\caption{Some men, possibly Dutch, looking at che
\label{figure:menandcheeses}
\end{figure}
```

which is the code we used to create Figure 3.3. You can probably see what's happening. The command `\includegraphics` gets your image, which can be PDF, PNG, JPG, GIF or PostScript. The `\begin{figure}... \end{figure}` code 'wraps' up whatever picture you're including, and allows L<sup>A</sup>T<sub>E</sub>X to treat it as an unbreakable **floating** thing that it will position for you as best it can in the document, while maintaining an overall nice typographical layout. This floating of figures can sometimes result in the figure ending up in a place you didn't expect, but in most cases L<sup>A</sup>T<sub>E</sub>X will make the most sensible choice. It's possible to employ finer control over figure placement, but that's beyond the scope of this introduction.

The `\includegraphics` command is not built in to core L<sup>A</sup>T<sub>E</sub>X but is in an additional **package**, which needs to be explicitly loaded. We load this package by using the command `\usepackage{graphicx}` in the document preamble, i.e. after the `\documentclass` command, but before `\begin{document}`. There are hundreds of such packages that extend L<sup>A</sup>T<sub>E</sub>X's functionality; the lab scripts you are reading now use over 30 of them.

### Exercise 3: Including images

Create a document `image.tex`, that contains some text (maybe from 'Lorem Ipsum'), together with a figure

containing an image of your choice, perhaps the one you made in Intro Lab 2 like Steve's Mr Noodle. There should also be a reference to the figure in the text. You should first use the program `convert` (check its man page) to create a PDF version of the image.



conve

### **Exercise 4: A reflection on your studies**

Your first proper  $\text{\LaTeX}$  document will be a very brief reflection on your studies so far.

Write a hand-crafted  $\text{\LaTeX}$  document called `courses-reflection` (exact name as usual please, otherwise `submit` will not work). It will have the following structure.

1. A table of contents.
2. A brief paragraph saying what the document is about.
3. A section, appropriately titled, for one of your course units, containing:
  - a brief paragraph saying what the unit is about.
  - A sub-section, appropriately titled, containing an enumeration of the three things you like the most about the unit.

## COMP10120 Report writing and Digital Typography

---

- A sub-section, appropriately titled, containing an enumeration of the three things you like the least about the unit.

A repeat of item 3 for each other unit you are studying. These sections should appear in alphabetical order by course code (e.g. COMP10120).

After completing and successfully compiling and viewing your document, you should spell-check it as follows.

```
ispell courses-reflection.tex
```

Once you are completely satisfied with it, produce a hard copy ready for marking.

### 3.4 Typesetting Mathematics

Earlier we saw this formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

It looks nice, doesn't it? To tell L<sup>A</sup>T<sub>E</sub>X to display this, you have to type a bit of magic, but it's easily-learned magic. To create such a formula in a WYSIWYG editor, there is also magic involved, but it usually involves

## COMP10120 Report writing and Digital Typography

---

a lot of mouse-clicking, and remembering special key combinations like 'control-this' and 'alt-that'. In  $\text{\LaTeX}$  it doesn't. You simply type this:

```
\[ x = \frac{-b \pm \sqrt{b^2-4ac}}{2a} \]
```

You can probably work out how most of this creates the formula, but it won't be obvious that the  $\[$  and  $\]$  symbols that enclose the formula mean 'typeset this as a displayed formula, giving it some vertical space from the surrounding text'. If we'd used  $\($  and  $\)$  instead to enclose the formula it would appear in-line, like this:  $x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$ . It still looks very nice, and observe how it's automatically been resized to fit, and that the lines of text have had their spacing changed a bit. This all looks simple, but the implementation inside  $\text{\LaTeX}$  and  $\text{\TeX}$  is complex. It involves **parsing** the description of the formula to create a corresponding tree data structure, which is then **recursively walked** to work out the horizontal and vertical typographical spacings needed. You'll meet these ideas in COMP11120 and COMP26120 Algorithms and Imperative Programming.

Here's another example, taken from COMP27112 Computer Graphics (it's a 'simple' **local illumination model** incorporating ambient, diffuse and specular reflection by multiple lights):

$$I = k_a I_a + \sum_{i=1}^M \frac{I_{p_i}}{d_i'} [k_d(\hat{N} \cdot \hat{L}_i) + k_s(\hat{R}_i \cdot \hat{V})^n]$$

We write this in L<sup>A</sup>T<sub>E</sub>X as follows:

```
\[ I = k_a I_a + \sum_{i=1}^M { \frac{{I_p}_i}{d_i'}
[ k_d(\hat{N} \cdot \hat{L}_i)
+ k_s(\hat{R}_i \cdot \hat{V})^n] \}
```

Try to match the L<sup>A</sup>T<sub>E</sub>X commands with the formula displayed above. You'll see lots of curly brackets, and this example illustrates their two uses in L<sup>A</sup>T<sub>E</sub>X. The first is to provide an argument to a command; for example `\hat{N}` means 'apply the `\hat` command to `N`', which creates  $\hat{N}$ , the vector  $N$  with a little hat on.

The second use of curly brackets is to group things together to avoid ambiguities. In the example you can see `\sum_{i=1}^M`, which creates a summation sign and its lower and upper limits:  $\sum_{i=1}^M$ . We wrap the lower bound, `i=1`, in curly brackets to group it into an indivisible unit. If we were to omit the brackets, writing `\sum_i=1^M`, L<sup>A</sup>T<sub>E</sub>X would then produce  $\sum_i = 1^M$ , which is not at all what we want (even L<sup>A</sup>T<sub>E</sub>X can't always know what we really want).

One final example. If we tell L<sup>A</sup>T<sub>E</sub>X:



```
\[ T_1 = \left[
  \begin{array}{cccc}
    \cos \theta & -\sin \theta & 0 & \delta \\
    \sin \theta & \cos \theta & 0 & \epsilon \\
    0 & 0 & 1 & \eta \\
    \alpha & \beta & \gamma & 1
  \end{array}
\right]
```

we'll get a splendid matrix which expresses a particular 3D geometrical transformation. (Don't worry if you don't recognise this, you'll be introduced to matrix notation in the latter part of COMP11120: Mathematical Techniques for Computer Science. For now you can just treat it as some mysterious maths):

$$T_1 = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & \delta \\ \sin \theta & \cos \theta & 0 & \epsilon \\ 0 & 0 & 1 & \eta \\ \alpha & \beta & \gamma & 1 \end{bmatrix}$$

In the  $\text{\LaTeX}$  code, `\left[` means 'big opening square bracket please'; `{cccc}` means 'an array with 4 columns please, with the items in each column centred'; `&` means 'start a new column'; and `\\` means 'start a new row'. You'll notice that  $\text{\LaTeX}$  knows about greek letters; it knows about most standard maths symbols too, and also the ways they're usually used.

There are some symbols that you might need (for example `\therefore`, which produces the usual three dots symbol  $\therefore$ ) that are not part of standard L<sup>A</sup>T<sub>E</sub>X. Many such symbols are provided by the `amssymb` package of symbols compiled by the American Mathematical Society. Details of the many symbols provided by this package can be found online at

[mirror.math.ku.edu/tex-archive/fonts/amssymb/doc](http://mirror.math.ku.edu/tex-archive/fonts/amssymb/doc)

To use these extra symbols, you need to have `\usepackage{amssymb}` in the document preamble.

### 3.4.1 A longer piece of mathematics

We conclude this section with an example of a longer piece of mathematics, that shows how you might typeset a complete mathematical argument rather than a single formula. The example is Euclid's proof of the fact that  $\sqrt{2}$  is irrational. It's an argument that has been covered in COMP11120 and is an example of proof by contradiction. It starts by making the assumption that it is *not the case* that  $\sqrt{2}$  is irrational, in other words that we can find integers  $p$  and  $q$  with  $\sqrt{2} = p/q$ .

We can assume, without losing anything, that  $p$  and

$q$  have *no common factors* (Why?)

$$\begin{aligned}\sqrt{2} &= p/q \\ q\sqrt{2} &= p \\ 2q^2 &= p^2\end{aligned}$$

This means that  $p^2$  is even, from which we can also deduce that  $p$  is even. (Why?)

This means that  $p = 2k$ , for some  $k$ , so ...

$$\begin{aligned}2q^2 &= (2k)^2 \\ \therefore 2q^2 &= 4k^2 \\ \text{so } q^2 &= 2k^2\end{aligned}$$

But this means that  $q^2$ , and hence  $q$  is even. Since  $p$  and  $q$  are both even we have a contradiction to our initial assumption, so no such  $p$  and  $q$  exist.

The way this is formatted is shown here:

It starts by making the assumption that it is `\emph{not the` that  `$\sqrt{2}$` is irrational, in other words that we can find integers  `$p$` and  `$q$` with  `$\sqrt{2} = p/q$`.

We can assume, without losing anything, that  `$p$` and  `$q$` have  `\emph{no common factors}` (Why?)

```
\[
\begin{array}{lcl}
\sqrt{2} & = & p/q \\
q\sqrt{2} & = & p \\
2q^2 & = & p^2
\end{array}
\]
```

## COMP10120 Report writing and Digital Typography

---

```
\]
```

This means that  $p^2$  is even, from which we can also deduce  $q$  is even. (Why?)

This means that  $p = 2k$ , for some  $k$ , so \dots

```
\[
\begin{array}{llcl}
& 2q^2 & = & (2k)^2 \\
\text{therefore} & 2q^2 & = & 4k^2 \\
\text{so} & q^2 & = & 2k^2
\end{array}
```

```
\]
```

But this means that  $q^2$ , and hence  $q$  is even. Since  $p$  and  $q$  are both even we have a contradiction to our initial assumption so no such  $p$  and  $q$  exist.

Notice the use of the `array` environment for getting things lined up. You can find much more information about this on the web.

L<sup>A</sup>T<sub>E</sub>X really shines at typesetting mathematics, and it would take pages and pages to describe all the features. Have a look at:

[en.wikibooks.org/wiki/LaTeX/Mathematics](http://en.wikibooks.org/wiki/LaTeX/Mathematics)

for a flavour.

### Exercise 5: Formatting mathematics

Now for an exercise that involves some mathematics. Please create a file `f-w.tex` which typesets the follow-

ing piece of text and mathematics.

There are many positive integer solutions to the equation

$$x^2 + y^2 = z^2$$

which can be rewritten as

$$z = \sqrt{x^2 + y^2}$$

For example (3, 4, 5) or (5, 12, 13). Such solutions are called *Pythagorean triples*.

However, for higher powers the situation is very different, and we have:-

**Theorem: Fermat-Wiles**

For all natural numbers  $n \geq 3$ , there are no integers  $x, y, z$  satisfying the equation

$$x^n + y^n = z^n$$

### 3.5 Some optional extras

#### Exercise 6: Exploring more L<sup>A</sup>T<sub>E</sub>X features (optional, but easy)

This exercise reinforces some of the things you've done already, but also introduces some features of L<sup>A</sup>T<sub>E</sub>X that

will be new to you. It is labelled as optional because it isn't marked but we would recommend that you all try it.

Take a copy of the file `/opt/info/courses/COMP10120/ex3/latex` and follow the instructions it contains. You should read it *before* you process it with  $\text{\LaTeX}$ .

### Exercise 7: A neat directory listing (optional, more challenging)

The fact that  $\text{\LaTeX}$  source files are just text makes it easy to have them generated by a program. You are going to exploit this and use  $\text{\LaTeX}$  to produce a neat document containing a listing of the files in any directory.

Find how to create tables in  $\text{\LaTeX}$  tables using the tabular environment. Make a  $\text{\LaTeX}$  document by hand that uses a table to neatly show the information about the files in some (small) directory (as produced by `ls -l`). Write a shell script, called `neat-ls`, that takes an argument which is the path name of a directory, and produces a  $\text{\LaTeX}$  document containing the above table for the given directory. The script could also process the  $\text{\LaTeX}$  and display the resulting `pdf` file. Finally, find out about the `longtable` package and change your script to use that instead.

### Exercise 8: A cleaning rota (optional, even more challenging)

Now you will write a shell script, called `make-rota`, that generates a house cleaning rota. (Very handy for next year ...). This is a table with columns such as Kitchen, Bathroom and Lounge. The rows will be weeks in the year, labelled with a date (e.g. always a Sunday). The idea is that house mates can write their initials in the boxes, promising to clean the corresponding room in that particular week. Start by making a sample table by hand.

Your script should take a single argument which is the date of the first day of the first row, e.g. `10/20/13` (the US format for 20 Oct 13). The result will be a document that contains as many rows as fit on a single page (find out by experimenting).

The dates of each row should be generated from the first. See the manual page for `date` to figure out how you can generate a date that is a number of weeks and days later than an existing one.

#### 3.5.1 Marking Scheme

You can view a copy of the marking scheme used by `submit` and `labprint` at

### 3.6 Next steps

$\text{\LaTeX}$  is a very powerful tool, and it does take a little getting used to. But it's not as scary as it might initially seem, and you should find that it doesn't take long to learn the common commands and techniques to make your documents well-structured and readable. As you become more comfortable with the basics and understand the principles of typesetting, finding out how to do more complex and sophisticated things is easy—there are hundreds of tutorials on the web.

Don't get carried away though, and use the features judiciously: just because you've found a cool new command doesn't mean that you should use it just to show off your  $\text{\LaTeX}$  skills: content and structure always trumps typographic twiddlings<sup>4</sup>.

---

<sup>4</sup>For example, footnotes like this are, for some reason, a really tempting feature to use in  $\text{\LaTeX}$ , and in this document we have abused them mercilessly to illustrate a point. Footnotes were originally a means for mechanical typesetters to insert comments, corrections or additions to existing documents without having to adjust the layout of a whole page, and of course in digital typography this is no longer necessary. They can justifiably be used for citations (though it's not common to do so these days) and URLs if you don't want these intruding into the body of your text. And in some very very rare cases, they can be used as an aside to the main flow of the text (which is what has been done in this document). But this should be used sparingly. . . . If something is important to say, put it in the main body of your text. If it's not, then perhaps you should think about leaving it out completely, or find some other way of including it. Asking



## 3.7 Acknowledgements

Figure 3.3 is from the Wikipedia article on [cheese](#).

## References

- C. Dickens (1852). *Bleak House*. Bradbury & Evans.
- D. Knuth (1999). *Digital Typography*. Cambridge University Press. ISBN: 1575860104.
- F. M. Liang (1983). “Word Hy-phen-a-tion by Com-puter”. PhD thesis. Department of Computer Science, Stanford University.
- Knuth, Donald E. and Michael F. Plass (1981). “Breaking paragraphs into lines”. In: *Software: Practice and Experience* 11.11, pp. 1119–1184. ISSN: 1097-024X. DOI: 10.1002/spe.4380111102.
- M. A. Tinker (1963). *Legibility of Print*. Iowa State University Press. ISBN: 978-0813824505.

---

the reader to look down at the bottom of a page to find out whether something is important or not is irritating, and as you’ve probably found out already, often means you lose track of your reading position. They also give the impression that the author isn’t clear about whether something is important or not, which is never good. And long footnotes like this one—while L<sup>A</sup>T<sub>E</sub>X will typeset them perfectly well—are just plain silly. Best avoid.

## COMP10120 Report writing and Digital Typography

- R. Bringhurst (1992). *The Elements of Typographic Style*.  
Hartley & Marks Publishers. ISBN: 0-88179-110-5.
- R. Frost (1924). *New Hampshire: A poem with notes and  
grace notes*. Henry Holt and Company.