Baskerville

The Annals of the UK TEX Users' Group **Editor: Sebastian Rahtz** **Vol. 8 No. 3**

ISSN 1354–5930 **July 1998**

Articles may be submitted via electronic mail to `baskerville@tex.ac.uk`, or on MSDOS-compatible discs, to Sebastian Rahtz, Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, to whom any correspondence concerning *Baskerville* should also be addressed.

This reprint of *Baskerville* is set in Times Roman, with Computer Modern Typewriter for literal text; the source is archived on CTAN in `usergrps/uktug`.

Back issues from the previous 12 months may be ordered from UKTUG for £2 each; earlier issues are archived on CTAN in `usergrps/uktug`.

Please send UKTUG subscriptions, and book or software orders, to Peter Abbott, 1 Eymore Close, Selly Oak, Birmingham B29 4LB. Fax/telephone: 0121 476 2159. Email enquiries about UKTUG to `uktug-enquiries@tex.ac.uk`.

Contents

I Snippets

1 TEX consultant sought

The Wellcome Trust is interested to hear from TEX experts who would be willing and able to provide technical support and consultancy to one user on an ad-hoc basis.

The project concerned aims to catalogue several hundred Arabic manuscripts and software used at the moment is MS Word, emTEX, emTeXgi and ArabTEX under MS Windows 3.11. This tool-set might change with an imminent move to Windows NT4 or could be changed if recommended. It is foreseen that there will be telephone inquiries as well as on-site visits.

Interested parties should apply to The Wellcome Trust by submitting a resume of their relevant technical experience, details of other commitments and expected payment mode. To find out more about the project, please e-mail Dr Nikolai Serikoff (`n.serikoff@wellcome.ac.uk`), to apply please send the requested information to Ms Dagmar Jeschin (`d.jeschin@wellcome.ac.uk` or Dagmar Jeschin, The Wellcome Trust, 210 Euston Road, London NW1 2BE) by the middle of August.

2 Committee members sought: a message from Philip Taylor

Having now served as Chairman of UKTUG for about nine months, I can without hesitation confirm that it is both an incredibly rewarding yet frequently frustrating experience. Rewarding, because at first hand I get to see how hard the various members of the Committee work to make UKTUG a better organisation; frustrating, because I see how much time gets wasted on the minutiae of committee work.

Yet despite the frustrations, work gets done: *Baskerville* is edited, formatted, printed and distributed; new editions of the TEX Live CD are prepared, pressed and sent out; and meetings and workshops are researched, planned, organised and announced. But who does this work, and why? The answer is 'the members of the Committee' (aided and abetted by willing volunteers such as Martyn Johnson of the Cambridge Computer Laboratory). And why? Only they can tell you! It's hard work, and the only reward is an occasional letter from a member expressing his or her thanks for the work which has been done.

But each year, a number of the Committee have to stand down. Some because they have served their time, and our Constitution (blessed be its name) ensures that no Committee member may serve more than his or her allotted time without standing down for at least one year. Some because pressure of 'real' work prevents them from spending the time on UKTUG activities that such activities demand.

And who is to replace them? That, dear Member, is where you come in! Do you care about TEX? Do you care about UKTUG? Do you have the energy to attend some four to six meeting a year, and to receive some two thousand or more electronic mail messages a year (and to send about 1/12 of that number yourself?). If you do, then this is the time to step forwards. A number of the present Committee will be standing down or retiring at the Annual General Meeting in September, and the remaining members of the Committee are very concerned to ensure that those leaving are replaced. If you are willing to stand, then *please* complete one of the nomination forms which accompanies this issue of *Baskerville*. Beg (on your bended knees, if necessary) two people to propose and second you. And canvass all your friends in UKTUG to vote for you, because this is going to be the hardest fought election in the history of UKTUG!

Do all this, and I look forward *very much* to seeing you on the Committee next year.

II Introduction to LaTeX for the rest of us

Malcolm Clark

I started my TeX-life with 'plain' TeX in about 1984. Being younger and more set in my ways, I truly believed that LaTeX was an affront to family values and represented yet another downward step in the sorry decline of moral and ethical values. If not actually the spawn of the devil, LaTeX users were at least beyond the pale. This view managed to sustain me for many years, until in 1989 Katherine Butterfield persuaded me to teach LaTeX to a research group at the University of California, Berkeley. LaTeX for the rest of us (or *LaTeX′* as it was then) was born. At first I thought (quite arrogantly) that I could merely adapt my existing TeX course to LaTeX. LaTeX requires a much more subtle and mellow thought process. Unlike TeX, where you really can do exactly what you want (whether or not it is a good idea), LaTeX requires that you relax and go with the flow. Trust it, and all will be well: remember LaTeX can smell your fear. Looking back, I have to say that the average dedicated LaTeX-er is a much less stressed individual than the average dedicated TeX-er.

When I published my *TeX Primer* in 1992, I planned to follow it shortly after by LaTeX for the rest of us. I was overtaken by the euphoria of LaTeX3, which eventually became the muted excitement of LaTeX 2ε, and had to start re-writing whole chunks. At the same sort of time, some excellent LaTeX books, like Goossens, Mittelbach & Samarin's *LaTeX Companion*, Kopka & Daly's second edition of their *Guide to LaTeX 2ε*, Goossens, Rahtz & Mittelbach's *LaTeX Graphics Companion*, and then the second edition of Lamport's own *LaTeX: A Document Preparation System* appeared on the scene. We were awash with good quality stuff (at last!). My enthusiasm for the project waned, but it was revitalised by the last course I taught, at the Technical University of Malaysia, where I had one of the biggest and best classes I have ever had the pleasure to teach. They helped me get things pretty close to their present state.

We are presenting this as a serialisation, though perhaps not quite in the same long and honourable tradition which includes Hardy and Dickens. What is in your hands represents about the first quarter of the text. The next quarter will be out later this year (we are aiming for *Baskerville* 8.5), and the other half should appear next year sometime, although precise dates and editions for 1999 are but figments of our collective imagination (neither the present editor nor myself are likely to be on the UK TUG committee next year, and our loyalty could be stretched. . .). There are also questions embedded in the text. Once all the text has been finished, solutions to these questions will be provided. This probably means yet another chunk of *Baskerville*, but since I haven't yet written the solutions, I don't know quite how much space they will take up.

All being well, and if our enthusiasm stands up, we're very tempted to create a pdf (Adobe Acrobat) version with tons of hyper-links. That would be available electronically (and maybe even as part of one of these magnificent TeX Live CDs that Sebastian Rahtz keeps producing). I love the idea of a hyper-LaTeX electronic book. The LaTeX source will of course go up as part of the *Baskerville* archive files on , where anyone may obtain it for their own pleasure and delectation.

There are too many people to thank to be able to do so individually, but I will always be grateful to Katherine Butterfield who provided the initial impetus. I gratefully acknowledge the many students from Santa Barbara to Johor Bahru who have been exposed to versions of the LaTeX courses I have given, and who have therefore unwittingly contributed to this project. I am especially privileged to have Sebastian Rahtz as editor, and David Carlisle as reviewer.

There will likely be errors in the text. I expect them to be errors of fact, omission, opinion or interpretation. I would be happy to hear of them, but except for errors of fact, I may choose to ignore them. Don't be surprised if I appear to lie or mislead in the initial chapters. This trait has a long and honorable tradition in books on TeX. Sometimes the truth takes a little longer to tease out. I've never believed that it was appropriate to tell all the truth in all circumstances. You won't find details of how to install TeX and LaTeX in LaTeX for the rest of us. This is quite deliberate since I believe it to be a non-problem, especially for members of this group, who all have a copy of the TeX Live CD.

LaTeX For The Rest Of Us

by

Malcolm Clark

Chapter 1. First steps

1 A brief history

LaTeX can be thought of as the fusion of two developments in computing software. One of these developments was TeX; the other was Scribe. Both are embedded in the academic world. Both came to fruition in the latter part of the 70's, and as such, preceded the introduction of personal computers. They also parallel the rise of large scale database systems, and the demise of traditional hand-set type.

1.1 TeX

TeX itself was developed as a typesetting tool by Donald Knuth of Stanford University [**?**]. Knuth wrote the first version of TeX in the late 1970s [**?**], primarily as a way of controlling the printed quality of his multi-volume work The Art of Computer Programming [**?**].

To some extent, and more particularly with the early versions, TeX tended to place some emphasis on the arrangement of the marks on the paper. After all, Knuth finishes [**?**] with the exhortation: 'GO FORTH and create masterpieces of the publishing art!'. He was clearly thinking in terms of marks on paper. Because of this attention to this relatively low level of detail, TeX is sometimes described as a *procedural* or *typographic markup* system.

1.2 Scribe

Scribe was written by Brian Reid at Carnegie-Mellon [**?**], and described in his thesis in 1980. In essence, Scribe ignored the content of the document, concentrating on the relationship of the parts: in a sense it imposed, or revealed, the implicit inter-relationships present. We may be prepared to acknowledge that a document is composed of (say)

- front matter
- main text
- annexed (or back) matter

and within each of these major divisions, we can identify subdivisions. The front matter might comprise

- title page
- author's dedication
- table of contents
- list of figures
- foreword
- preface
- acknowledgements

and so on. Some of these elements may be optional. This presents structure as very hierarchical, but it is possible to construct arrangements which are not so fiercely arranged, and share features with other, less formal, types of organisation.

One of the major points to make is that Scribe was a system of *descriptive* or *logical markup*: it described the structure of the document. It made no comment about the arrangement of the words on the page. In fact, there need be no 'page', in the sense of 'paper' page. Changing the characteristics of the 'carrier medium' changed nothing about the document structure. While it is possible to disagree with extreme versions of this viewpoint [**?**], it did offer a powerful and attractive way to separate documents from the worries of formatting.

1.3 LaTeX

Leslie Lamport [**?**] took the ideas of Scribe and the typesetting capabilities of TeX and fused them into LaTeX, a piece of software where an author could include an account of the structure together with the content – the words. Lamport sought to devise a system where the author need know nothing about the details of typesetting – justification, choice of typeface, page breaking, ligatures, and so on. Merely by saying that the document was a 'book', or a 'report', the text would be formatted appropriately.

In contemporary jargon, TeX is described as the *formatting* or *typesetting engine* whose presence need never be revealed. The author concentrates on the content, merely guiding the structure gently by noting features like sections, subsections, figures, etc. To this Lamport added or adapted tools to sweep up tables of contents, lists of figures, cross referencing, bibliographic control and indexing. Perhaps these *value added* tools are the features which still give LaTeX the edge over almost any other software of this type.

That and the fact that it is available, if not freely, certainly very cheaply. A great deal of the work on TeX and LaTeX has been done through the public domain. This does not mean it was done for free: a great deal of money was spent, one way or another, in perfecting these tools. Fortunately for the rest of us, a lot of it was 'public' (i.e. taxpayers') money, rather than 'commercial' money. Even the commercial versions of TeX and LaTeX are very cheap when compared to 'comparable' commercial alternatives.

LaTeX absorbed all TeX's mathematical capability. Since LaTeX is written in TeX (which itself has some pretensions to being a programming language), anything that is possible in TeX can also be done in LaTeX, so we have lost no functionality. This is really only theoretically true, since, as we may see, TeX itself has some memory limitations: once the LaTeX functionality is loaded, there is limited space left for other things. But this is an architectural problem rather than a conceptual one.

One of Lamport's advantages was that he was able to work with Knuth, as Knuth was developing the 'final' version of TeX. Certain requirements which came to light with the development of the early versions of LaTeX were incorporated into TeX82, the version of TeX which we all used until Knuth [**?**] made some relatively small changes in 1989; this latter change was a relatively straightforward and painless upgrade which may even have gone unnoticed by some. Lamport is quoted somewhere as saying that he could not have achieved what he had done without the active cooperation of Knuth. This is not to minimise Lamport's achievement, since LaTeX is a wholly remarkable piece of software. It has defects, but to a large extent these turn out not to be too limiting. Some of its problems are really problems inherent in TeX. But more of that later (perhaps).

1.4 LaTeX 2ε

Some other changes have to be documented. For a long period, LaTeX was also known as LaTeX 2.09. This actually covered an incremental refinement where bug fixes were gradually incorporated without changing the version number and was leading to uncertainty of what was the 'current' version of LaTeX, and some divergence of what it actually constituted. This was not altogether satisfactory. By 1990 there was the beginning of a proposal to change LaTeX substantially [**?**], for a variety of sound reasons. Ultimately this gelled into a proposal for a new version to be called LaTeX3 [**?**]. In the meantime, to hold us until the new version is ready, a consolidation, LaTeX 2ε was publicly released in 1994. This is the version of LaTeX which constitutes the core of this discussion. From time to time it will be necessary to recap some of the history which brought us to LaTeX 2ε, but by and large the discussion is directed at LaTeX 2ε. When some peculiarity of LaTeX 2ε is mentioned it will be referred to as such, but most of the time, LaTeX and LaTeX 2ε will mean the same thing.

When will LaTeX3 be released? Or rather, will your investment in learning LaTeX be lost when LaTeX3 appears? I hope not, not least since my investment will be lost too. It will take some time before LaTeX3 appears, and even when it does, the concepts will likely stay much the same, although there may be detail changes [**?**]. Furthermore, it is likely that there may be some sort of compatibility mode, where existing (or *legacy*) documents written in LaTeX will be capable of being processed with LaTeX3. Nothing should be wasted. The investment is just too vast to risk.

1.5 A proviso

One possible disadvantage of the LaTeX/Scribe approach is that it requires that you have a notion of just what it is you are trying to do: some people write electronic or digital documents in a very haphazard, piecemeal fashion, jumping around the document, and basically losing track of things. If this is the way you work, and you resent the rigour of structure, seeing it as a straight-jacket rather than scaffolding, you may be unhappy with LaTeX. A little bit of discipline may help to refine your ideas, and, in the longer term, achieve more.

It is certainly true that you will not learn all there is to know about LaTeX from this brief account. On the other hand, with its aid, you may learn enough to be able to produce (apparently) complex documents, and without too

much trouble. Systems which are learned quickly sometimes lack stamina. To some extent LaTeX can be approached in an 'intuitive' way, but no-one is likely to try to claim that it is especially 'user-friendly', at least, not until you are properly introduced.

2 A note on pronunciation

Some people get very worked up about the pronunciation of TeX and LaTeX. Knuth says of 'TeX'

> "Insiders pronounce the χ of TeX as the Greek chi, not as a 'x', so that TeX rhymes with the word blecchhh. It's the 'ch' sound in Scottish words like *loch* or German words like *ach*; it's a Spanish 'j' and a Russian 'kh'."

Unfortunately, few of the English-speaking world are equipped to pronounce *loch*, and the German-speaking world has (at least) two pronunciations of *ach* – the less obvious, southern German form, is like *ash* or perhaps *asch*. In the end, the majority of English-speakers seem to end up with *tecks* or *teck*.

Lamport sagely notes:

> "…pronunciation is best determined by usage, not fiat. TeX is usually pronounced *teck*, making *lah*-teck, lah-*teck*, and *lay*-teck the logical choices; but language is not always logical, so *lay*-tecks is also possible."

About the only possibility Lamport does not cover is *L-A-teck*, so perhaps we can conclude that that is not a recommended pronunciation.

Of course, Knuth's description, as given here, begs the question a little, since it does not say where the Greek chi comes from. Knuth's original notion, in naming the software TeX, was to use the roman form of $\tau\varepsilon\chi$, which is the beginning of the Greek word transliterated as *tekhnē*, which we use in words like *technology*. To the Greeks this encompassed art *and* technology. A better rendition into contemporary english would probably be something like *craftsmanship*. An exposition of *tekhnē* might be the Parthenon in Athens, or even any piece of work which was the responsibility of Isambard Kingdom Brunel.

But, however we pronounce 'TeX' and 'LaTeX', we will always recognize the logos, which are so difficult to produce in most every other system. If you do have to write TeX or LaTeX in a system where you cannot drop the 'E' or raise the 'A', write them like 'TeX' and 'LaTeX'.

QUESTION 1.1 *What would you say were the structural elements of a memo? or a letter? What are the bits which are always present?*

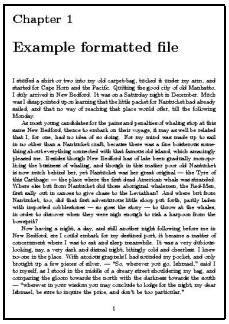
QUESTION 1.2 *This question cannot be answered immediately, but it may be revealing if you keep it in your mind as you read through the book. Ponder on the extent to which developments in software are related to available hardware. Try to imagine the sort of hardware and operating systems which were available as Knuth was developing TeX. Contrast that with the hardware and operating systems which have led to the development of* wysiwyg *(what you see is what you get) software. What might Knuth and Lamport have done if they were developing TeX/LaTeX today?*

QUESTION 1.3 *Take a word processing or document formatting system with which you are familiar – `nroff`, Runoff,* Pagemaker, *WordPerfect, Word, FrameMaker, Interleaf, Quark Xpress, lout,* HTML, *paper and pencil, typewriter, etc., and note down what seem to be their* descriptive *and* procedural *elements. Which bits are concerned only with positioning of characters on the page, and which bits describe items independently of their form? There may even be a third category where the positioning and the 'content' are combined.*

3 Let's try it

Since LaTeX was developed for documentation, it is a good idea to start with a fairly straightforward document and see what it is we have to do in order to translate it onto the page. We will assume that 'marks on paper' are the goal. The 'paperless office' still seems as far away as ever, despite efforts from most computer manufacturers, Xerox PARC, and the damage being done to the forests of the world.

We will begin with the output: the end product of the document preparation process. This end product, Figure 1Let's try itfigure.18, is not spectacular, but it does have some interesting features, most of which we probably take for granted. If we examine the input file, we will see that the text is enclosed between the instructions in the following manner:



Chapter 1

Example formatted file

I stuffed a shirt or two into my old carpet-bag, tucked it under my arm, and started for Cape Horn and the Pacific. Quitting the good city of old Manhatto, I duly arrived in New Bedford. It was on a Saturday night in December. Much was I disappointed upon learning that the little packet for Nantucket had already sailed, and that no way of reaching that place would offer, till the following Monday.

As most young candidates for the pains and penalties of whaling stop at this same New Bedford, thence to embark on their voyage, it may as well be related that I, for one, had no idea of so doing. For my mind was made up to sail in no other than a Nantucket craft, because there was a fine boisterous something about everything connected with that famous old island, which amazingly pleased me. Besides though New Bedford has of late been gradually monopolizing the business of whaling, and though in this matter poor old Nantucket is now much behind her, yet Nantucket was her great original — the Tyre of this Carthage; — the place where the first dead American whale was stranded. Where else but from Nantucket did those aboriginal whalemen, the Red-Men, first sally out in canoes to give chase to the Leviathan? And where but from Nantucket, too, did that first adventurous little sloop put forth, partly laden with imported cobblestones — so goes the story — to throw at the whales, in order to discover when they were nigh enough to risk a harpoon from the bowsprit?

Now having a night, a day, and still another night following before me in New Bedford, ere I could embark for my destined port, it became a matter of concernment where I was to eat and sleep meanwhile. It was a very dubious-looking, nay, a very dark and dismal night, bitingly cold and cheerless. I knew no-one in the place. With anxious grapnels I had sounded my pocket, and only brought up a few pieces of silver, — "So, wherever you go, Ishmael," said I to myself, as I stood in the middle of a dreary street shouldering my bag, and comparing the gloom towards the north with the darkness towards the south — "wherever in your wisdom you may conclude to lodge for the night, my dear Ishmael, be sure to inquire the price, and don't be too particular."

1

Figure 1.

```
\documentclass{book}
\begin{document}
\chapter{Example formatted file}
I stuffed a shirt or two into my old carpet-bag,
tucked it under my arm, and started for Cape Horn
and the Pacific. Quitting the good city of old
Manhatto, I duly arrived in New Bedford.  It was
on a Saturday night in December. Much was I
disappointed upon learning that the little packet
for Nantucket had already sailed, and that no way
of reaching that place would offer, till the
following Monday.

As most young candidates for the pains and
penalties of whaling stop at this same New
Bedford, thence to embark on their voyage, it may
as well be related that I, for one, had no idea of
so doing. For my mind was made up to sail in no
other than a Nantucket craft, because there was a
fine boisterous something about everything
connected with that famous old island, which
amazingly pleased me. Besides though New Bedford
has of late been gradually monopolizing the
business of whaling, and though in this matter
poor old Nantucket is now much behind her, yet
Nantucket was her great original --- the Tyre of
this Carthage; --- the place where the first dead
American whale was stranded. Where else but from
Nantucket did those aboriginal whalemen, the
Red-Men, first sally out in canoes to give chase
to the Leviathan?  And where but from Nantucket,
too, did that first adventurous little sloop put
forth, partly laden with imported cobblestones ---
so goes the story --- to throw at the whales, in
order to discover when they were nigh enough to
risk a harpoon from the bowsprit?

Now having a night, a day, and still another night
following before me in New Bedford, ere I could
embark for my destined port, it became a matter of
concernment where I was to eat and sleep
meanwhile. It was a very dubious-looking, nay, a
very dark and dismal night, bitingly cold and
cheerless. I knew no-one in the place.  With
anxious grapnels I had sounded my pocket, and only
brought up a few pieces of silver, --- ''So,
wherever you go, Ishmael,'' said I to myself, as I
```

```
stood in the middle of a dreary street shouldering
my bag, and comparing the gloom towards the north
with the darkness towards the south --- ‘‘wherever
in your wisdom you may conclude to lodge for the
night, my dear Ishmael, be sure to inquire the
price, and don’t be too particular.’’
\end{document}
```

LaTeX has handled all the details of page size, line-breaks, page-breaks (take my word on that one), hyphenation, ligaturing, paragraph indentation, justification, the white space between lines, and so on. These are all implied through the adoption of the ‘book’ document style.

Another feature which may have appeared transparently is the text’s division into paragraphs. If you leave a blank line, LaTeX will implicitly assume that this represents a paragraph break. And by default it will leave indentation at the beginning of the paragraph.

4 Elementary typography

What are ligatures? In many ways they are left-overs from the days of scribes. They are recognized ‘running together’ of letters, to form a new symbol. The most common examples, found in many typefaces are:

ff → ff
fi → fi
fl → fl
ffi → ffi
ffl → ffl

This is not simply a ‘very close’ duo or trio of individual letters, but a completely new symbol.

This is by no means a universal transformation. It is bound by typeface, by culture, and by time. Gutenberg’s original 42 line Bible, printed in about 1455, has at least 50 different ligatures. He was trying to emulate the work of scribes. They had evolved all sorts of shortenings which he strove to adopt. By the 19th century, English had standardized on the 5 or so above, but ‘ct’ and ‘st’ ligatures persisted at least up to the middle of this century, although in rather specialist typefaces. Although ligatures are more common in ‘serifed’ typefaces, some sans serif faces do have ligatures. Equally some fairly traditional serifed faces have abandoned them. Stanley Morison, who is credited with the design of the ‘Times’ typeface, may well be rotating in his grave now that The Times has dispensed with ligatures.

LaTeX takes care of ligatures without even thinking about it too deeply. It also takes care of ‘kerning’. Kerning is a much more recent introduction, and occurs where letters overlap, but do not touch. This is much more noticeable in capital letters: OXO or WAVE ought to be good examples, where the horizontal extent of each character overlaps with that of the next character. Again, kerning is not supported in all fonts. In particular, you would probably not want kerning in a font which was supposed to look like typewriter characters. Kerning relates to the moving together of characters to make the letter spacing more pleasing: it might be better to describe this as ‘less displeasing’. We may notice when a book is difficult to read because of the typography: we seldom notice when it is easy to read. This is as it should be. A warning: exposure to TeX and LaTeX can affect your enjoyment of the printed word. You start to look at the typography much more closely. It can destroy some books entirely.

You may perhaps notice some other things which are done entirely automatically, and which tend to distinguish ‘typesetting’ systems from ‘desktop publishing’ or word processing systems. The first distinguishing feature is the three different sorts of horizontal rules or dashes, used in text (although in Figure 1Let’s try itfigure.18 only two of them are used):

In fact, dashes come in lots of forms. Formally we identify the following: the hyphen, the en-dash, and the em-dash (and the minus sign too, but we’ll look at that later): see Table 1Elementary typographytable.20. In most typestyles these will be different characters. A hyphen is fairly obvious, and is conveyed to LaTeX as a `-` symbol. An en-dash is a longer symbol (about the length of an N in the current font), and is therefore conveyed to LaTeX as `--`; an en-dash is usually employed to convey the idea of a range, for example 1–10. Occasionally too, an en-dash is used when two names are joined, like Runge–Kutta or Russell–Hertzsprung, although they are often just hyphenated. Thirdly, the em-dash is even longer (related to the capital M), and is given to LaTeX as `---`. The em-dash is punctuation in text. (The minus is a mathematical symbol which has to be given in maths – to be covered later – as `$-$`.)

'correct name'	LaTeX form	typeset form	example
hyphen	-	-	hy-phen
en-dash	--	–	1–7
em-dash	---	—	Knuth—the archiTeXt
minus	-	–	$x - y$

Table 1. Length and meaning of horizontal lines

Another feature is treated by LaTeX as a sort of ligature. Quotation marks (i.e. double inverted commas) come in open and close varieties in many fonts. LaTeX employs the quote (also known as *apostrophe* and *prime*) and grave. On one of my keyboards 'quote' is on the right of the keyboard, just beside the `return` key). The left quote, or *grave* is at the top left of the keyboard, beyond the numerical characters). Unfortunately, the 'extra' characters on a keyboard do not have standardized positions, and the location on your keyboard may differ. Clearly, this provides you with only a single inverted left or right comma at a time. LaTeX 'ligatures' a pair of inverted commas, whether they be grave or quote, to form a single 'double' quote mark. Thus, to form a double open quote mark on output (the '66' form) – " – type a pair of graves – ``, and to form the corresponding ('99') close double quotes – " – type a pair of quotes – ''. The double quote symbol on your keyboard – " – should not be used.

5 In the beginning

The content of *every* document we write with LaTeX has a beginning and end delimited by the `\begin{document}` – `\end{document}` pairing. Note that this is the beginning and end of textual content of the document. It is not the beginning and end of the file of information. While it seems that there is little point in placing anything after the `\end{document}` statement, since it will never by seen by LaTeX, it is somewhere to keep notes, odd ideas and even pieces of text you plan to use in the future.

This is described as a 'document' *environment*. The notion of an enclosed environment, with a matching `\begin` and `\end` pair is fairly fundamental to LaTeX.

The very beginning of the file of information has a `\documentclass` statement. This statement will be modified by additional instructions which will determine many of the layout alternatives which will be adopted; it also has some bearing on the sorts of structures which are available. LaTeX has a small number of fixed document *classes* associated with it.

6 Classes of style, flair and panache

The basic LaTeX document classes are `book`, `report`, `article`, `letter`, `slides`, `proc` and `ltxdoc`. The first four of these were the original 'styles' released with LaTeX and documented (to some extent) in the original manual [**?**]. The remainder were introduced with LaTeX 2ε, when the concept of 'document class' was also introduced, and separated carefully (though not wholly convincingly) from 'style' (see also [**?**]). In general the names are descriptive, meaningful and obvious. For those already familiar with LaTeX the class `slides` replaces SLiTeX, while `proc` is intended to handle a conference proceedings. The last, `ltxdoc` is an indirect product of the documentation effort related to LaTeX 2ε and LaTeX3, and is intended to assist in the documentation of classes and styles. It will not be examined here.

There are other classes around, many of which are referred to and described in the LaTeX Companion [**?**]. In theory, these classes and styles should be documented in what Lamport refers to as the Local Guide. He had the notion that every installation would have a locally produced Guide which would contain useful information on the availability of fonts, classes, styles, support software, and so on. Guides are usually conspicuous by their absence. The Companion is an admirable substitute.

The implications of each style are sometimes hard to grasp. Just how does `report` differ from `article`? To some extent this is to misunderstand the function of the classes and the model of *declarative markup* which was outlined earlier. Lamport worked with a number of document designers to develop the classes/styles. And he made it a non-trivial job to 'tweak' the classes/styles to adapt them. In other words, he felt (very strongly) that you, as the user, should get on with what you were good at, namely writing, and let LaTeX get on with what it was good at, namely formatting. To be fair, LaTeX 2ε does simplify some modes of modification. Everybody wants to tweak. And not everybody enjoys the layouts adopted by Lamport. They certainly come out of a particular typographic tradition,

\documentclass	principal options
book	twocolumn, 10pt, 11pt, 12pt, twoside
report	twocolumn, 10pt, 11pt, 12pt, twoside
article	twocolumn, 10pt, 11pt, 12pt, twoside
letter	10pt, 11pt, 12pt
slide	
proc	twocolumn, 10pt, 11pt, 12pt, twoside
ltxdoc	twocolumn, 10pt, 11pt, 12pt, twoside

Table 2. Some of the default range of classes and their options

size options	paper size
a4paper	210 mm × 297 mm
a5paper	148 mm × 210 mm
b5paper	176 mm × 250 mm
letterpaper	8.5 in × 11 in
legalpaper	8.5 in × 14 in
executivepaper	7.25 in × 10.5 in

Table 3. The range of page size options

one which is not universal. It is one possible comment that the original LaTeX book does not use the book layout available with LaTeX.

To use a particular class, we say

```
\documentclass{report}
```

or whichever of the classes we wish to choose. In addition, there are *options*, which modify the fundamental class. For example, there are 11pt and 12pt options which allow us to change the basic size of the font used in the main body of the text. By default, LaTeX uses a 10 point font for its 'body text'. *Options* precede the *class*, and are enclosed in square brackets:

```
\documentclass[11pt]{report}
```

QUESTION 1.4 *The example in Figure 1Let's try itfigure.18 was created using the* book *class (since it is really a chapter from a book). To get some crude notion of the effects of changing styles (and options), you could edit the* \documentclass *statement to create* article*s or* report*s at 10, 11 or 12 point.*

QUESTION 1.5 *What other document classes do you require?*

Although there are few document classes, there are far more options. There are also lots of 'packages' which allow us to change some aspect of the default presentation and features of LaTeX. Changing an option or package is far easier than tackling a class. Later we may look at the modifying or creating packages to alter the default behaviour of LaTeX.

Clearly the options 10pt, 11pt and 12pt are mutually exclusive. A document may have only one basic type size. But there are other options which affect some other aspect of the presentation. For example, twoside formats the output for printing on both sides of the page ('duplex' printing). Of course this does not mean they will come out of the printer printed on both sides. It simply affects the physical position of the mass of text on odd and even pages, so that when they are used as masters (in a photocopier, for example), the text will fall in a position which ensures that there is no 'show through': that is, you cannot see the other page's text as a 'shadow' through the paper, since it is obscured by the text on the page you are reading. Such subtle niceties are the traditional concern of printers. Another option is twocolumn, which produces two-column output. When we specify several options, they are separated by commas:

```
\documentclass[twocolumn,11pt,twoside]{report}
```

QUESTION 1.6 *Try a few options. Does* twoside *work correctly on your printer? Be bold.*

There are also a set of built in page sizes, Table 3Classes of style, flair and panachetable.26, which do two things: they set the height and width of the text itself, and they also position where the text will be placed on the page. Obviously LaTeX cannot know what page size you will ultimately use, and selecting a4paper will not guarantee that the output device, perhaps a laser printer, is actually loaded with the correct size of paper. If you do not use any of these size options, the default is letterpaper, a rather distressing result outside the US. If you were using some other size of paper (perhaps for a book), it would be necessary to set the width and height of the text through some other means. This method is likely to be inappropriate. It is also possible to add another option to these, landscape, which swaps the height and width to give a 'landscape' orientation rather than the more normal 'portrait'.

7 More information

Part of the structure of a document will be the 'front' matter: things like the title, the author, and so on. LaTeX allows you to include this information too. We can write something like

```
\title{The Carpet-Bag}
\author{Herman Melville}
\date{1851}
```

By itself, this does nothing. Until we say `\maketitle`, this information does not appear on the page. Since we would expect a title to appear at the beginning of a document, the `\title`, `\author` and `\date` should normally appear before we try to write out the title. The 'best' place for the information is therefore between the `\documentclass` specification and the `\begin{document}` statement. From time to time this location will be described as the document *preamble*. But to be truthful, it can go anywhere. If you do not specify a `\date`, LaTeX will use the current one (which happens to be Friday 4þ July 1998). In order to omit the date entirely, you could type `\date{}`. The empty braces are necessary: `\date` by itself would not give you an acceptable result.

```
\documentclass[twocolumn,11pt]{book}
\title{The Carpet-Bag}
\author{Herman Melville}
\date{1851}
\begin{document}
\maketitle
.
.
\end{document}
```

QUESTION 1.7 *Go ahead, do it. Of course you do not have to use* `twocolumn` *or the* `book` *document class.*

8 Some back-tracking

All along we have ignored the presence of some key characters. I have presented instructions like `\documentclass` and `\begin` without much fanfare. The backslash character, '`\`', is of major importance. All LaTeX instructions are introduced by such a character. In a sense, it is the TeX/LaTeX 'escape character'. Whatever follows a `\` is treated in a special way, and will not appear on the page as such. Obviously there are ways of making backslashes appear on the page, since this document was produced through LaTeX. Note too that LaTeX is case-sensitive. `\DocumentClass` will not be understood. In very general terms, the majority of the instructions you supply specifically for LaTeX will be in lower case. Only a few instructions use upper case characters.

A LaTeX instruction may be constructed in *only* one of two ways. The first, most flexible way, which we have used already, is by a backslash followed by an arbitrary number of alphabetic characters. Thus we might expect to see the following as legal LaTeX instructions:

```
\documentclass  \raggedbottom  \section
\maketitle      \vspace        \hspace
\kill           \sloppy        \newpage
\framebox       \line          \circle
```

Each one of these is a legitimate instruction which already exists in LaTeX. The alternative form is a backslash followed by a *single non-alphabetic* character. Thus we might expect to see the following:

```
\~  \\  \.
\1  \#  \
```

Note that 'space' is a legitimate non-alphabetic character. Since it is sometimes difficult to see when 'space' is meant, from time to time we will use ␣ to indicate a space which occurs between words and/or instructions; when `\`␣ is used it indicates the instruction where a space is the non-alphabetic character (In LaTeX this instruction actually generates a space).

We do not find instructions like

```
\A4   \half-sized  \up&down
\1.1  \(longer)    \m.clark
```

On the other hand, the *option*s may be made up of mixtures of alphabetic and non-alphabetic characters (as we have already seen: `11pt` or `A4paper`). Options are always enclosed by the square brackets.

The *argument* to an instruction, is the part in braces (or *curly* brackets). An instruction with an argument *always* requires an argument, although as suggested earlier, the argument could be 'null' – `\date{}`. Sometimes you may only specify one of a handful of alternatives, as in the case of the arguments of `\documentclass`, while in other cases, like `\author`, you have complete freedom. There are also instructions which require no argument at all – `\maketitle` is an example. The braces are never printed by LaTeX. Of course, there are ways to print `{` and `}` or even } and {, but we shall look at them later.

QUESTION 1.8 *What does happen when you specify a non-existent document class, or a non-existent option? Hint: try it. Advice: Don't Panic (in large friendly letters).*

9 Trying to correct mistakes

If you did the last exercise, you will have deliberately made mistakes on input, and will have encountered LaTeX's error processing capability. It is possible to correct errors as LaTeX is running interactively. If you should successfully correct the input in this way, you must also remember to correct the original (assuming you might just need to re-run sometime). To be more truthful, you may have met both LaTeX's error processing, *and* TeX's. Neither is especially elegant.

One of the most common mistakes is to invoke an instruction which does not exist – either because you let intuition take the upper hand and assume that the instruction must exist, or, more likely, because you mis-spell it. LaTeX will object:

```
! Undefined control sequence.
l.3 \start
          {section}{Start}
?
```

LaTeX is trying hard to indicate where the error lies, principally by breaking the line to indicate just where it has foundered. In the example above, it is the instruction `\start`. At this point it is wise to note that (LA)TeX refers to a 'control sequence', a piece of jargon which is often replaced by the word 'command', and in this text by 'instruction'.

Note however that we are left with a `?` prompt. Let's be intuitive, and assume that LaTeX has something up its sleeve, like additional help. How do you get help? How about typing '`h`'? Lo, something else appears.

```
The instruction at the end of the top line
of your error message was never \def'ed. If
you have misspelled it (e.g., '\hobx'),
type 'I' and the correct spelling (e.g.,
'I\hbox'). Otherwise just continue,
and I'll forget about whatever was undefined.

?
```

Well, who can understand that? Not very helpful at all. But we've still been left with a question mark. Let's try '`h`' again.

```
Sorry, I already gave what help I could...
Maybe you should try asking a human?
An error might have occurred before
I noticed any problems.
''If all else fails, read the instructions.''
```

This is one of (LA)TeX's attempts at humour or light heartedness. All the same, it is probably good advice.

In response to the '`?`' prompt, you have several options. You may type any of the following:

? where (LA)TeX gives a summary of the following options:

```
Type <return> to proceed, S to scroll
future error messages,
R to run without stopping,
Q to run quietly,
```

```
I to insert something,
E to edit your file,
1 or ... or 9 to ignore the next 1 to 9
tokens of input,
H for help, X to quit.
```

<return> just prod the `return` key (or the `enter` key). TEX proceeds as best it can, until it encounters another error.

X or x LATEX stops (eXits); you are returned to the operating system. Any pages which have already been completed may not be lost, but the current one will certainly be lost; the previous one might be as well.

E or e this stands for 'edit', and should drop you into an editor. Not all implementations have this linking of LATEX and editors. If you do not drop into an editor, you will simply be returned to the operating system prompt. The best way to find out if it does work on your system is to try it. If it does work you would find yourself editing the erroneous file, at about the right line – or rather, where LATEX thinks the error occurred. Often it is remarkably close to the error.

I or i you may now type text to be Inserted at the current place in input. At first this seems intimidating, but with some practice it does become a viable route. Its major drawback is that you tend to forget these 'dynamic' corrections.

a number between 1 and 99 LATEX deletes this number of characters and instructions from input. The characters or instructions are those which are waiting to be read – in other words, LATEX has not yet 'seen' them, or tried to do anything with them. LATEX then asks for more information (you could insert, etc.): LATEX sees an instruction as a single item: it also sees a character as a single item – otherwise termed 'token' – for example, `\textbf{bold}` is seen as the 7 'tokens' `\textbf`, `{`, `b`, `o`, `l`, `d` and `}`. A space is treated as a token, *except* when it immediately follows a instruction.

H or h (LA)TEX gives some sort of help.

S or s this is like typing `<return>` (or `<enter>`) for every subsequent error message. The error messages are logged, but you have no chance of interaction.

R or r this is like `S`, only worse; under no circumstances stop.

Q or q even even worse; LATEX suppresses all output to the terminal (goes a lot faster, subjectively), but perhaps not the best route unless you are very confident that you know what you are doing, which obviously you don't, else you would not have made a mistake in the first place. There is an instruction equivalent, `\batchmode`, which gives a good clue of when this would be most commonly used.

There is obviously a temptation to just type `<return>` and let LATEX surge ahead to report on any other errors. Unfortunately the corrections LATEX may have made in order to do something apparently sensible may lead to other mistakes later on. When I don't feel up to mental gymnastics I much prefer to leave LATEX (by typing `x` or `e`), correct the error, and then return to LATEX.

10 Hints

Don't panic. Error messages are often difficult to fathom, and it can be easier to solve the problem with reference to your text and the instructions which were included in the file than to attempt to understand what the error message is saying. But do not ignore the messages entirely. Sometimes they can be uncannily accurate and helpful.

What are the common errors? Failing to balance braces (every open brace must have a corresponding close brace). This is really quite difficult to correct through the error reporting and correction procedure outlined above. If you have a brace open, LATEX may do quite a lot of work before you realise that something has gone wrong. In such a case it is almost impossible to put things back together in a reasonable way. Sometimes this situation manifests itself when processing finished and you see a message that

```
(\end occurred inside a group at level 1)
```

In a similar way, from time to time you might fail to '`\end`' something you have '`\begin`ned'; again the same message may appear. More readily apparent, you may `\begin` something but `\end` it with the wrong thing.

Mis-types are a frequent source of problems. After all, if you type `artical` for `article`, LATEX can hardly be expected to divine your intention. Any instruction which is presented to LATEX 'incorrectly' will present a problem. It really does not care about your own spelling. It does care about the instructions you give it, and since it thinks you know best, it attempts to follow your orders to the letter.

Apart from that, what can go wrong?

Chapter 2. More things you should know

11 Changing font

Once desktop publishing arrived[1] people expected to change font at the drop of a hat. Brought up on typewriters, with no typographic knowledge at all, they suddenly acquired a whole new vocabulary of 'Palatino', 'Bookman', 'Zapf Chancery', and strongly held opinions about kerning, letterspacing, serifs, tracking and other arcane typographical subjects (rational views may be found in [**?**] and [**?**]).

12 Some history (again)

When Knuth designed TeX, he also designed fonts to go with it. Or rather, he used an existing typeface, Monotype's Modern 8A, and produced what he termed 'Computer Modern'. He had some help, but basically this 'family' is the suite of fonts with which TeX (and LaTeX) were initially tuned. That is not to say you cannot use others, just that some work *may* be needed before it will be a success. The Computer Modern family comprises about 75 'different' fonts. Most families comprise three or four different fonts – medium, bold, italic and so on. Now, 75 fonts does not mean that they are all strikingly and immediately different: Knuth took the notion of a 'design size' rather seriously. What this means is that the font is designed to be displayed and read at a specific size. Thus we have several versions of Computer Modern Roman, designed to be read at 5, 6, 7, 8, 9, 10, 12, 17 and 25 points [**?**]. These are genuinely different: the proportions change subtly as we change size.

Contemporary digital fonts tend to be stored as 'outline' information, where to arrive at a particular size, we 'merely' magnify the 'prototype' outline. This will not change the outline, just the size. Adobe encoded 'hints' into their POSTSCRIPT 'Type 1' outlines, which help preserve the subtle changes. MicroSoft's 'TrueType' fonts contain similar hints – but distinct enough to avoid legal action. However, these hints have as much to do with accommodating digital fonts to the various resolutions at which they may be displayed as to modifying them for the point size at which they are displayed.

Returning to the Computer Modern fonts, they are usually stored as 'rather compressed' bit maps, for a given resolution. Thus a laser printer would typically require resolutions of 300 dpi or 600 dpi, and a phototypesetter perhaps 1270 dpi. Outline fonts clearly have an advantage here, since you would only need one font to be stored, The rasterization of an outline takes place at print time, in the printer's own processor. At the time, Knuth's solution seemed a good one. It certainly assured quality, but it does require a lot of storage space for all those bit maps. But disk space is cheap. And if you can store the bit maps down on your laser printer or phototypesetter, the data transmission times can be reduced. If you are working on a network, you may have to store the bit maps on a server, where you then have the time consuming business of shipping them down to the printer.

If we delve a little more into the process, we discover that Knuth also created a 'language' to describe fonts – METAFONT [**?**]. The descriptions of Computer Modern are encoded in METAFONT (see, for example [**?**]). Some implementations of LaTeX will include METAFONT, and you may find that when you need particular fonts, they are generated on the fly. Typically, you will take many years to require all the Computer Modern fonts at all the sizes possible, and you can afford to accumulate them over time, METAFONT generating them as required. Since some of these fonts may only ever be required in rather exceptional and infrequent circumstances, a common strategy is to delete the less well used ones after use. Given the current processor speed commonly available, this is quite a sound approach. But it does depend on you having a suitably configured system where all these various components can inter-communicate.

Equally, Postscript and Truetype versions of the Computer Modern fonts are now generally available, both commercially and within the public domain.

For many years TeX and LaTeX gained a reputation of working with only Computer Modern and unfortunately, some people, especially publishers, did not like Computer Modern, usually preferring Times. The reputation was unfounded, but it stemmed from two main origins: firstly, it did take some time and effort (and sometime access to proprietary information) to utilise other fonts; and secondly, other fonts did not necessarily have the full range of characters which

[1] Desktop publishing arrived long before Paul Brainerd of Aldus invented the term and Apple created the bandwagon. Xerox PARC had been doing all that about 1978 with the Bravo system – even to the extent of on-demand immediate laser printing. TeX users throughout the known world had been publishing via their desktop terminals, and the UNIX world had also been using the `nroff`/`troff` family and its pre-processors. The key to success is marketing.

were readily available for Computer Modern. Eventually the problem was solved quite comprehensively by Frank Mittelbach and Rainer Schöpf with the introduction of the 'New Font Selection Scheme' [**?**] or NFSS. Although we will look at this later, if you are anxious to eschew the use of good old Computer Modern for your particular favourite, consult Chapter 7 of the Companion.

13 What fonts do we have?

We won't actually use all the fonts here, but just look at the ones we get by default with LaTeX.[2] Depending how you count, there are eleven types or 'styles' of font available, divided into three independent groups: the 'family' group is Roman, Sans Serif and Typewriter: the 'series' group comprises Medium and Bold: while the 'shape' group is Upright, Italic, Slanted and Small Capitals. That leaves one orphan, Normal. Normal is the style used in the 'body' of the text. But even then that only comes to ten. The eleventh is an 'emphasis' style. The results of using emphasis is to change the appearance in some way. Note that we do not use underlining at all – that is a typing convention for emphasis which is never[3] used in typesetting. We are in the Gutenberg tradition, not the Sholes tradition. Table 4What fonts do we have?table.42 shows what these fonts look like on the page. If you look closely you will note that there appears to be no difference between Upright, Medium, Roman and Normal. At this stage, this is true; later we shall see the distinctions in more detail.

QUESTION 2.1 *Sholes who? By now, very few people have direct experience of having used a typewriter. For those who do, what other leftovers from the days of the typewriter afflict us? Many word processing packages defer to typewriting practise and may provide hints and clues. Is it necessary to ask who Gutenberg was? or Caxton? or the cutely named Wynkyn de Worde? or even Baskerville?*

To change the type 'style', we use the simple, fairly mnemonic instruction given in Table 4What fonts do we have?table.42. LaTeX has a very powerful technique available which makes life easy – grouping. Grouping is a notion central to the whole existence of LaTeX. We have already met some sorts of grouping, where we have a `\begin` and `\end`. There is a simpler grouping – `{` and `}`. If we, for example say

```
The  song really \emph{is} \textit{A-Sitting
On A Gate}: and the tune's my own invention.
```

the parts enclosed in braces are the parts which will be emphasised (by `\emph`), or turned into italics (by `\textit`). The remainder of the passage will use whatever happens to be the font already in use. The word `is` is really 'emphasised', while the `A-Sitting On A Gate` part is a title (of sorts – see Table 5What fonts do we have?table.43). We may wish all such titles to be represented in a particular font or style, to assist readers distinguish this sort of information.

It is a good idea to review the reasoning behind changing font: presumably it is to provide the reader with visual clues to the structure of the contents of the document. Thus headings are usually in a different font from the text, and different levels of heading usually have slightly different characteristics: quoted text is often presented in a different font, and so too is emphasised material. Given that human perception seems to work best with 7 ± 2 discrete items at a time, you can see that you probably do not want too many different fonts at a time.

At last, this brings in the use of emphasis: LaTeX uses the instruction `\emph` to denote emphasis. Let's look again at the last example to see that

```
The  song really \emph{is} \textit{A-Sitting
On A Gate}: and the tune's my own invention.
```

gives

The song really *is A-Sitting On A Gate*: and the tune's my own invention.

Here we have distinguished the two elements 'emphasised text' and 'italicized text'. In this circumstance, it is hardly visible, except in the marked up text. But let's change things a bit:

```
\textit{The  song really \emph{is} \textit{A-Sitting On A Gate}:
and the tune's my own invention.}
```

This time we find that the emphasised text is in the roman font:

[2] These notes were originally written with the intention that the font used would be Computer Modern. The house style of this august organ is Baskerville. If the NFSS transformations have been done correctly, all should be well.

[3] hardly ever

Style	instruction	example
Roman	`\textrm`	The quick brown fox comes to the aid of the Hamburgerfons
Upright	`\textup`	The quick brown fox comes to the aid of the Hamburgerfons
Sans Serif	`\textsf`	The quick brown fox comes to the aid of the Hamburgerfons
Typewriter	`\texttt`	`The quick brown fox comes to the aid of the Hamburgerfons`
Medium	`\textmd`	The quick brown fox comes to the aid of the Hamburgerfons
Bold	`\textbf`	**The quick brown fox comes to the aid of the Hamburgerfons**
Upright	`\textup`	The quick brown fox comes to the aid of the Hamburgerfons
Italic	`\textit`	*The quick brown fox comes to the aid of the Hamburgerfons*
Slanted	`\textsl`	The quick brown fox comes to the aid of the Hamburgerfons
Small Capitals	`\textsc`	THE QUICK BROWN FOX COMES TO THE AID OF THE HAMBURGERFONS
Normal	`\textnormal`	The quick brown fox comes to the aid of the Hamburgerfons

Table 4. font styles readily available

what the name of the song is called	Haddocks' Eyes
what the name is	The Aged Aged Man
what the song is called	Ways and Means
what the song really *is*	A-sitting On A Gate
the tune	I give thee all, I can no more

Table 5. Carrolliginian confusion – totally irrelevant here

The song really is *A-Sitting On A Gate: and the tune's my own invention.*

LaTeX is smart enough to do that, but unfortunately is not smart enough to change the title information. After all, you have specified that it is italicised. The fact that you then repeat the italicisation instruction changes nothing, although you may have wished that the second `\textit` instruction changes to some different font so that that part looks different. Nevertheless, this can be a useful feature. Do not think of 'emphasis' as simply 'italics', because that is not what it means. It is a function which may be implemented in a variety of ways, depending on the circumstances.

QUESTION 2.2 *Take some of the earlier text material and change fonts, either over the whole document, or just selected parts. Which fonts do you find easier to read? Are the fonts which are easier to read on the screen also the fonts which are easier to read on paper?*

14 Bigger or smaller

You will have observed that fonts are available in more than one size. LaTeX has a series of instructions which allow you to change the size of any font, very easily. The instructions used are given in Table 6Bigger or smallertable.46. Their exact behaviour depends on the options you have set up. The order of the sequence remains constant, but sometimes two adjacent 'sizes' may use the same sized font (without telling you). The immediate question is 'are all these sizes available for all the fonts?'

size	style
`\tiny`	Aa
`\scriptsize`	Aa
`\footnotesize`	Aa
`\small`	Aa
`\normalsize`	Aa
`\large`	Aa
`\Large`	Aa
`\LARGE`	Aa
`\huge`	Aa
`\Huge`	Aa

Table 6. the sizes available

QUESTION 2.3 *Are all these sizes available for all the fonts? Remember to try the different size options as well as the different fonts.*

The way that these size-changing instructions are used is quite different from the font-changing instructions. A font changing instruction looks like `\textrm{text}`, while to change size you say `{\small text}`. This difference is quite crucial. The size changing merely indicates a new condition which is operational until the end of the group in which it occurs. We could, for example, say

```
\Huge Come, \huge I'll \LARGE take \Large no
\large denial, \normalsize  we must
\small have a \tiny trial
```

and each change in size would take place when the instruction occurs in the text:

Come, I'll take no denial, we must have a trial

We might be advised to enclose the whole thing in braces, lest any succeeding text was also placed in the `\tiny` size of font. If the size changes are not grouped, separating them out, they remain in effect.

This different style of usage harks back to the older form of LaTeX. It should not cause too much confusion, since you would not normally want to change size in your text. Any such size changes should be restricted to special conditions, like section headings or new environments, which you would not be writing as you go along, but would have been defined separately. Later we may see how to do these.

15 Accumulation

It may come as a pleasant surprise to realise that it is possible to combine the font changing and the size changing mechanisms, and that something sensible happens. For example,

```
a \textsf{\Large mouses's} \textit{\tiny tail}
```

will result in 'a mouses's *tail*', which is presumably the effect we wished.

Sometimes fonts are not available in all sizes: had you written

```
a \textsl{\large mouses's} \textsc{\tiny tail}
```

you may find there is no `\tiny` small capitals font available. LaTeX will substitute something else – and does so without telling you. To some extent this lack of a suitable font is implementation dependent, since it can be possible to create or obtain the 'correct' font for use in this situation. In the current conditions it would yield:

a mouse's TAIL

Do note that these size changing instructions are absolute, not relative. Saying something like

```
a {\small mouses's {\small tail}}
```

will not make the tail even smaller. In passing, note that if there is any chance of the text appearing on more than one line, it would be best to end a paragraph (by inserting a blank line) before the closing `}` or else you may get small text on the existing line spacing. Or if you were using `\large`, the even worse case of large text on the existing line

instruction	explanation
`\ss`	gives the German ß
`\OE`	gives the Œ diphthong
`\oe`	gives the œ diphthong
`\AE`	gives the Æ diphthong
`\ae`	gives the æ diphthong
`\O`	gives the letter Ø
`\o`	gives the letter ø
`\AA`	gives the letter Å
`\aa`	gives the letter å
`\L`	gives the letter Ł
`\l`	gives the letter ł
`?‘`	gives the symbol ¿
`!‘`	gives the symbol ¡

Table 7. the so-called ‘national’ characters

spacing, with ungainly and uneven line spacing as LaTeX moves things around to fit. But you really ought not to be changing size as you go along like this. The examples here are hardly the stuff of the normal article, report or book.

In the previous example, we would not be able to obtain a slanted small capital font by

```
a \textsl{mouse’s {\tiny \textsc{tail}}}
```

But if we return to our description of fonts in three groups, a family, a series and a shape, we will find that these three are truly indpendent. That is to say I can ask for a bold sans serif **mouse’s tail** by

```
\textbf{\textsf{mouse’s tail}}
```

or for a typewriter medium small caps `mouse’s tail` by

```
\texttt{\textmd\textsc{{mouse’s tail}}}
```

This is an example that fails in the current font set up. If you examine the log file closely enough, you will find that you have been told, and also told what substitute has been chosen. This is a topic we’ll look at in more detail later.

In principle, you may combine any family with any series with any shape and expect to see something sensible.

QUESTION 2.4 *Try some of these combinations. How would you make the mouse’s tail progressively smaller?*

16 Accenting the positive

Now it is time for something even more frivolous. One of the nice features of LaTeX (although of marginal real use in English), is its excellent support of diacritical marks and foreign letters. Naturally there is an ulterior motive for introducing these now.

First the special letters. LaTeX recognises instructions for the diphthongs Œ, Æ, œ and æ (commonly used in Latin and in some Scandinavian languages, among others). It also recognises the German ‘ß’ (ess-zet) symbol. It will handle the Å, å, Ø and ø of some Scandinavian languages. And lastly, it copes with the Polish suppressed-L, Ł and ł.

How do we get these into our text? Follow the `\` by a special instruction as shown in Table 7Accenting the positivetable.51. How do you use these new instructions? The ‘recommended’ way for beginning LaTeX-users is to enclose them in braces. This never fails. Thus to write Œdipus in LaTeX, you actually write `{\OE}dipus`. This helps to distinguish `{\OE}dipus` from `\OEdipus`, which LaTeX would assume was a new (probably unknown) instruction.

There is another, briefer, way, which you will encounter frequently: leave a blank space between the instruction and the rest of the word

```
\OE dipus
```

Note that in this form any extra spaces between `\OE` and `dipus` will be ignored, as far as creating the output is concerned. Since all extra spaces are ignored, leaving a few extra, or even writing

```
\OE
dipus
```

will not leave a space between the diphthong and the rest of the word which follows when the passage is set. In other words, a line break (a 'carriage control') is just another space to LaTeX. This gobbling up of extra blanks is a normal feature of LaTeX. If we group the instruction, there is no need to follow it by a space, and in fact `{\OE} dipus` will give the result 'Œ dipus', which is not what was required.

QUESTION 2.5 *In what ways are the following different?*

```
{\OE }dipus, {\OE} dipus, \OE {dipus},
\O{E}dipus
```

And what other 'useful' possibilities are there?

What implication does this shorthand method have for instructions which come at the end of words? Consider trying to write:

```
the Schlo\ss of the Rhine valley
```

The word 'Schloß' would appear as we require, but the spaces which follow would be ignored, and the next word would begin immediately after the ß. This is generally not what we want. In order to solve this problem, there are a variety of solutions. We could have used `{\ss}` (the grouped instruction), as recommended, or we can use a new instruction, `\␣`, that is, the backslash followed by a space, which introduces a 'command space' (or a 'control space' or even a 'hard space'). Thus what we probably wanted was

```
the Schlo\ss\ of the Rhine valley
```

In this approach, the 'hard space' lets LaTeX know where the instruction ends. If you look at this more closely, you will realise that there are other ways to signify the end of a instruction. One of them was illustrated with the `{\OE}dipus` sequence. There the `}` was able to indicate the end of the instruction. Actually, `\OE{}dipus` would have had the same effect. The sequence `{}` looks odd, and seems to mean nothing, but from time to time, even nothing has its uses. In the alternative shown with the `Schlo\ss\␣` sequence, the occurrence of a `\` 'obviously' begins a new instruction, and therefore indicates the end of the previous one.

There is an advantage to using the shorthand, which is not readily apparent: the kerning information is used. Once we group the instructions, the kerns between characters are ignored. While it is unlikely that you will notice this in many situations (especially if you are not familiar with French or German typography), it is perhaps an encouragement to use the shorthand. There is always some pleasure to gained from the feeling that you are somehow doing things 'right'.

In Table 7Accenting the positivetable.51 the two 'inverted' symbols were included as a sort of national character. These are a little odd, and have been known to cause confusion from time to time. If you are careless where you type spaces and confuse your opening and closing quotes, you could end up with something like this dreadfully forced example:

```
....oh no!‘he said quite emphatically....
```

These two national characters need not be enclosed in braces, since they are not standard LaTeX instructions, in the sense that they do not begin with the backslash. They work rather differently. On the other hand, feel free to put braces in if you want. Nothing untoward will occur.

LaTeX also has lots of diacriticals (Lamport refers to them as accents, but some are not). The list is given in Table 8Accenting the positivetable.53. By and large, the instructions are fairly logically named.

In order to get accents over i and j, you really ought to take the dot off first. LaTeX supports a dotless i and j, provided by `\i` and `\j`. These allow you to do things like î (from `\^{\i}`), or even íj, (from `\t{\i\j}`), should you ever find a reason to do so.

The general rule with all these sequences is – accent first, then letter. At first this sounds counter intuitive, after all, we say 'e-acute', or 'o-circumflex'.

None of these accents are really 'fundamental' to LaTeX, in the sense that they are all created in much the same way, a way that is accessible. If we knew enough we could even create our own diacriticals. For example, Polish has a ů character, where the 'ring' can be defined as a diacritical symbol.

QUESTION 2.6 *In order to demonstrate your skills with these fancy fripperies, try setting some of the following:*

`\‘`	grave	`\‘{e}` gives è
`\’`	acute or aigu	`\’{e}` gives é
`\^`	circumflex or hat	`\^{o}` gives ô
`\v`	inverted circumflex (háček accent)	`\v{c}` gives č
`\u`	breve	`\u{o}` gives ŏ
`\=`	macron, long vowel	`\={u}` gives ū
`\"`	umlaut or dieresis	`\"{u}` gives ü
`\H`	Hungarian umlaut	`\H{o}` gives ő
`\~`	tilde	`\~{n}` gives ñ
`\.`	dot accent	`\.{y}` gives ẏ
`\t`	tie	`\t{oo}` gives o͡o
`\c`	cedilla	`\c{c}` gives ç
`\d`	dot under	`\d{d}` gives ḍ
`\b`	bar under	`\b{a}` gives a̲

and although not diacriticals, we should mention

`\i`	dotless i	`\i` gives ı
`\j`	dotless j	`\j` gives ȷ

Table 8. the diacriticals

Hsän Tsang, Tathāgata, Śākyamuni, Vaiśravana; Pi-ma-wên, Li Yüan-chi, Ānanda Káśyapa, Manjuśrī; Väinämöinen, Äijö, Völuspá; Anne Brontë, Honoré de Balzac, François Rabelais; naïve, régime, façade, manœuvre, encyclopædia; Ægean Sea, Château d’If, Gdańsk, Öőz, Nîmes.

Zde se všemožně snaží mě přeluvit, abych ještě několik měsía napsal ještě jednu operù. Hayır! İş öyle değil. Büyüğü küçüğüne takilmayı pek severdi. Ce fût d’ores et déjà une idée dégénérée et ambigüe.

If you were normally writing in a language which made use of diacritical marks or national characters, you would find this all rather tedious. You would likely have a keyboard which had characters like (for example) é, å, ß, and so on. Having to insert the special LaTeX sequences would, at the very least, be error-prone. It is possible to tailor LaTeX to take account of this, so that when you type the single character é, LaTeX interprets it correctly. Later we’ll look at some LaTeX ‘packages’ which simplify this task. There should be added benefits, since it will ensure that the kerning information is also handled correctly; and the hyphenation should also be correct for the language you are using.

The default ‘language’, which LaTeX assumes it is using is American English. Not only does this mean that other languages are hyphenated in a rather haphazard way, it also results in words with diacriticals generated through the instructions of Table 8Accenting the positivetable.53 being hyphenated even less well (no hyphenation after the first diacritic). This is a deliberate gloss which does not quite tell the truth. We’ll revisit this topic later.

17 Hy-phen-a-tion

LaTeX chooses hyphenation points by a hybrid method. It uses an algorithmic technique which is supplemented by a small dictionary of ‘exceptions’. The algorithm it uses can be taught to recognise the appropriate hyphenation points for other languages (see Appendix H of [**?**]). A large number of alternatives have been collected or developed by Johannes Braams [**?**] and are available as a ‘package’ to LaTeX users. If you really have to hyphenate German, or Esperanto, or even English now, refer to Chapter 9 of the Companion.

You can hyphenate words yourself as they occur, by inserting a special command which indicates a *potential* or *discretionary*. For example, in `Tyr\-rhenian` the `\-` indicates a discretionary hyphen. Note that this instruction is not followed by a space. Declaring each potential hyphenation is tedious, and one alternative is to declare the potential hyphenations as:

```
\hyphenation{Tyr-rhenian manu-script manu-scripts}
```

That is, simply a list of hyphenated words, separated by spaces. This has a global effect, since what happens here is that these words are added (temporarily) to LaTeX’s ‘exception dictionary’. As noted above, TeX/LaTeX hyphenates

by algorithm, but there is a small dictionary of exceptions. The best place for this hyphenation instruction is in the preamble.

Note that LaTeX will not realize that `manu-scripts` is merely a regularly formed plural of `manuscript`. Similarly, this mechanism can know nothing about any other regularly formed inflections. On the other hand, the standard hyphenation can cope with many inflections. Explicitly declaring hyphenation points will not help words which already contain hyphens. While 'pricking' by itself can be hyphenated to 'prick-ing', 'pin-pricking' is not hyphenated to 'pin-prick-ing'. If we were to say

```
\hyphenation{pin-prick-ing}
```

we could easily end up with 'pinpricking'. In this case we would have to use `pin-prick\-ing` throughout the document.

QUESTION 2.7 *One way to see the effect of hyphenation is to reduce the number of words per line. The easiest way you know so far is to use a large font size, or to use the* `twocolumn` *option. Choose some text and do so.*

Chapter 3. Environmentally aware

18 Environments

LaTeX is environmentally aware! One of the key features of LaTeX is the way in which it uses 'environments': that is, an 'enclosed' structure (which may be part of another structure), which has some kind of unity. We have already met one environment, the `document`. There are a number of other LaTeX environments, all intended for slightly different purposes.

Environments have the common feature that they are bounded by a `\begin{..}` `\end{..}` pair. You may only ever terminate an environment by an environment of exactly the same name. However, the `\end{document}` will generally close everything, perhaps with a few oblique comments from LaTeX.

19 Quoting

To illustrate an environment, look first at quoting. LaTeX has two slightly different environments for quoting: `quote` and `quotation`. A `quote` is intended for short, perhaps one line, quotations.

```
\begin{quote}
Joe Bob says check it out.
\end{quote}
```

yields

> Joe Bob says check it out.

What do we note about this? The quote is indented to offset it from the existing text, and there is a little white space above and below. If we had a longer quotation, we would be able to see whether the indentation we see was a 'normal' paragraph indentation or whether the whole quote was indented or whether the whole quote is indented:

> I am not now, and never have been, a member of the American Communist Party – *attributed to Mao Xedung*.

Obviously it applies to all lines.

The other alternative is `quotation`, which is really designed for longer quotations containing several paragraphs:

> I know your works, that you are neither hot or cold: I would that you were either hot or cold.
>
> But because you are lukewarm, neither hot or cold, I will spit you out.

which may be obtained from

```
\begin{quotation}
I know your works, that you are neither
hot or cold:
I would that you were either hot or cold.

But because you are lukewarm,
neither hot or cold, I will spit you out.
\end{quotation}
```

QUESTION 3.1 *Use the `quotation` environment to produce a piece of quoted text. If your mind is a complete blank, you could always try:*

```
It's one hundred and six miles to Chicago;
we got a full tank of gas, half a pack of
cigarettes, it's dark, and we're wearing
sunglasses.

Hit it.
```

20 Verse or worse

Lamport introduces another environment which he describes thus:

> Poetry is displayed with the `verse` environment.

Few poets would want to have their work described as verse, and even fewer would use the `verse` environment to present it. However, the environment is not without its uses. And in fact, to be 'fair' it has some rather appealing qualities. Unlike every other situation we have seen so far, LaTeX asks that you do the line breaking yourself in the `verse` environment. Do this by terminating a line with `\\`. As we shall discover, this works in all sorts of other places too.

Bring me my Bow of burning gold:
Bring me my Arrows of desire:
Bring me my Spear: O clouds unfold!
Bring me my Chariot of fire!

was produced from

```
\begin{verse}
Bring me my Bow of burning gold:\\
Bring me my Arrows of desire:\\
Bring me my Spear: O clouds unfold!\\
Bring me my Chariot of fire!
\end{verse}
```

One feature may be worth emphasising. The last line of a stanza does not have to end with `\\`. Since it is immediately followed by `\end{verse}` there is no need to introduce an additional (and therefore redundant) instruction.

QUESTION 3.2 *Find out what happens when you end the last line with* `\\`. *Does LaTeX handle it in an elegant and friendly way?*

Lamport clearly had Blake in mind when he created the `verse` environment, since lines which are too long for the page width wrap over nicely, with a little bit of extra indentation:

Excess of sorrow laughs. Excess of joy weeps.
The roaring of lions, the howling of wolves, the raging of the stormy sea, and the destructive sword, are portions of eternity too great for the eye of man.
The fox condemns the trap, not himself

There are two other features of note here. The first is that the `\\` instruction has a variant form, the `*` form. If you write `\\*`, LaTeX will not allow the page to be broken at that point. This useful feature is an example of a variant form for the instruction. We will find that many LaTeX instructions have a `*` variant, which modifies the default way in which the instruction is handled.

The second feature is that the instruction may take an optional argument. We have met optional arguments before, and we shall meet them again and again. Optional arguments are always enclosed in square brackets – `[ ]`. Here the optional argument allows us to specify how much additional space to leave between lines – a *dimension* is specified, complete with its units. In the case of the `\\` instruction we would have something like `\\[0.5cm]` where some dimension, in this case 0.5 cm, was specified in the square brackets. This is similar to the optional arguments we have already met for `\documentclass`, except that `\documentclass` also requires an argument in braces. Once we know what units we can use, this optional argument offers some useful possibilities.

21 Units – a digression

We have come a long way, without any real reference to the sizes of things, except fonts. In general, we shouldn't be too concerned with sizes of most things on the page, since the selection of the document style implicitly specifies many of the relationships and sizes of parts of the document. Nevertheless, there are times when we want to do something particular, where we must somehow specify a size.

Table 9Units – a digressiontable.64 shows the units which LaTeX 'knows' about, together with their conversions. To specify a dimension, say `12pt`, `0.7pc` or `12in`, or even `12 pt`, `0.7 pc` or `12 in` if you find that easier to write or read. If you inadvertently say `12inches`, LaTeX will assume that the `12in` is the dimension, and that the `ches` is text to be printed. Of course `12points` wouldn't even be understood and would generate an error.

(La)TeX uses the conversions as *exact* ratios – in other words, even if they are not the exact and true conversions, they are what LaTeX uses (this ensures a degree of standardisation within LaTeX). Any value preceding one of these units may be specified as either a whole (integer) number, or one with a decimal point (decimal numbers would not

abbreviation	name	conversion
pt	point	
pc	pica	1 pc=12 pt
in	inch	1 in=72.27 pt
bp	big point	72 bp=1 in
cm	centimeter	1 in=2.54 cm
mm	millimeter	10 mm=1 cm
dd	didôt point	1157 dd=1238 pt
cc	cicero	1 cc=12 dd
sp	scaled point	65536 sp=2^{16}sp=1 pt
em	width of 'M'	(current font)
ex	height of 'x'	(current font)

Table 9. Units used in LaTeX to specify dimensions

make much sense in the case of 'scaled points'). LaTeX does not support mixtures of units, so that 'two picas and four points' would not be acceptable, except as 2.3333333pc or 28pt.

You do not have to stick to any particular dimensions throughout your document. Mix metric, Imperial and printers' dimensions as you wish. The printer's point makes some sort of sense as a base unit, since all the fonts are described in terms of their 'point' size. But outside that particular area, any other 'size' can be described in any of the above dimensions. If you do not include a dimension, LaTeX will often assume points are meant; it is safer to give the dimension. If the dimension is zero, it still needs units.

The `em` and `ex` deserve a little more explanation. They are the only two 'relative measures'. In typographic terms, an em (also known as a *mutton*) is a horizontal measure (being approximately the width of a capital 'M' in the *current* font), while an ex (also known as an *en-nut*) is a vertical measure (the height of an 'x' in the *current* font). If you change font, the values of em and ex will also change. Although an em is intended as a horizontal measure, it will be possible to use it in a vertical sense: similarly, the ex may be used for horizontal amounts. LaTeX is not that picky.

21.1 The point

In this sort of situation, in verse, it is more general to specify a distance which is related in some way to the font in use: a relative rather than an absolute figure:

Our plesance here is all vain glory,
This fause warld is but transitory;
The flesh is brukle, the Fiend is slee:

Timor mortis conturbat me.

The 'refrain' is separated from the rest of the poem by an additional 2 ex, obtained by terminating the line preceding the refrain by `\\[2ex]`:

```
The flesh is brukle, the Fiend is slee:\\[2ex]
```

QUESTION 3.3 *Try that last example, but use different units. How would you leave a gap which was equivalent to a blank line?*

22 Lists

If LaTeX is remembered for nothing else, it will be remembered for the way in which Lamport elevated the 'list' into a common, powerful, all-pervading structure. The more I think about it, the more I see myself surrounded by documents which are most effectively presented as lists, or lists within lists. As a technique for structuring an argument, and giving relative weights to various parts and sub-parts, it can be enlightening and fundamental.

LaTeX has three basic list structures, each with slightly different attributes and visual properties. There are also facilities to extend these. The structures are

- enumerate

- itemize
- description

Of course, this is a list! It was generated by

```
\begin{itemize}
\item enumerate
\item itemize
\item description
\end{itemize}
```

Now we have (almost) the whole syntax[4] of these list-making environments.

As usual, the environment must be terminated with the correct name.

QUESTION 3.4 *Try doing something like this:*

```
\begin{itemize}
\item just a word or two
\item and another one
\end{enumerate}
```

The error message which LaTeX gives in this context is very annoying:

```
! \begin{itemize} on input line 3 ended
  by \end{description}.
```

It is informative that LaTeX stops at this point, but also helpful that it will allow you to continue. A pity it misleads.

If you are an `emacs` guru, or have another similarly powerful editor[5], and skill, you could ensure that your `\begin{..}` was always terminated correctly.

Itemizing

Using the `itemize` environment, each item is marked in some way. Different levels of itemization are given different symbols. This implies that you can nest itemization instructions. In fact, you can nest the whole concept of environments in general, and lists in particular[6].

QUESTION 3.5 *Find out what the symbols are for the various levels of itemization.*

QUESTION 3.6 *Find out just how deeply you can nest itemization.*

Enumerating

As you may guess, the `enumerate` environment numbers each item. This implies that there must be a counter marching away somewhere, incrementing each time `\item` is encountered. And so there is. There may be several, since we can, as usual, nest enumeration too.

QUESTION 3.7 *Find out what happens when you nest levels of enumeration. For example, you could use one of your last exercises, changing the* `\begin{itemize}` *to* `\begin{enumerate}`*. Remember to change the* `\end` *too.*

QUESTION 3.8 *Now nest enumeration within itemization (or even itemization within enumeration).*

[4] You should be able to guess that there are probably some optional arguments lurking about somewhere. There usually are.

[5] Correctly, `emacs` is a way of life. Like (LA)TEX, there is nothing `emacs` cannot do, in the right hands.

[6] Almost true; don't go too deep.

Describing
In the `description` environment, you *must* provide an optional (*sic*) argument on the `\item`. This argument will be the 'key' to the list. In the previous list environments, LaTeX generated something (a symbol, or a number/letter) to mark each `item`. This time, you have to provide the 'mark':

```
\begin{description}
\item[aardvark] mainly harmless --
as in ''aardvark never hurt anyone''.
\item[Betty Boop] a pleasure and a
delight. One of \emph{the} great artistes.
\item[crow] family of birds including the
magpie (\textsl{pica pica})
\end{description}
```

which comes out looking like

aardvark mainly harmless – as in "aardvark never hurt anyone".
Betty Boop a pleasure and a delight. One of *the* great artistes.
crow family of birds including the magpie (pica pica)

QUESTION 3.9 *Perhaps I exaggerate when I say that this environment requires an optional argument. Demonstrate what happens if you omit the optional argument on the* `\item`*s in* `description`*. Or whether the other list environments may take optional arguments.*

QUESTION 3.10 *It has always seemed to me that these three basic list environments are not well named – with two verbs and a noun. Devise a better and more consistent set of names.* HTML *uses the rather terse* `ol` *(ordered list),* `ul` *(unordered list), and* `dl` *(data list).*

23 Read my lips

Another two pairs of environments may be useful. As these notes show, it is often useful to be able to express chunks of text in typrewriter form – that is 'verbatim', where every symbol is recorded as it was typed in. This may not be a frequent requirement. LaTeX has a `verbatim` environment which will handle this. Obviously it has to turn off lots of LaTeX's normal processing, and as a result there are some limitations. When the environment is ended, the `\end{verabtim}` must occur on a line by itself. A variant is the *-form, where any blanks (spaces) within the material will be printed out as ␣. Besides needing to be able to present chunks of text literally, it may also be necessary to do this for odd symbols or words. The command `\verb` is provided for this. For example, if we write

```
The backslash, \verb+\+, introduces every instruction, with a few
exceptions, including \verb=~=.
```

then the result will be

The backslash, \, introduces every instruction, with a few exceptions, including ~.

Note how `\verb` works. The material which is to be expressed verbatim is 'enclosed' within a pair of repeated arbitrary symbols. The only symbols which may not be used are `*`, since there is a variant, `\verb*`, where any enclosed blanks are again represented by ␣. For example

```
The tag \verb*+< html>+ will not be
recognized, since the space (\verb*+ +)
after the \verb+<+ should not be present
```

would give us

The tag <␣html> will not be recognized, since the space (␣) after the < should not be present

Although line endings are respected in the `verbatim` environment, you should not have a line end in the body of a `\verb`. This is seldom a problem, but if you use an editor which wraps (like *Textures*), you may inadvertantly break a `\verb` over a line. LaTeX will note the error. Another important factor is that neither of these instructions may appear in the argument of another instruction.

The sister environment to `verbatim` is `alltt`. This is identical, except that `\`, `{` and `}` all have their 'normal' meanings. We may therefore have

```
\begin{alltt}
Even in text we need to  \textsl{slant}
\end{alltt}
```

to yield

```
Even in text we need to  slant
```

If we wish to use this feature we should first ensure that the `alltt` package is loaded. We'll cover this aspect a little later.

24 One paragraph at a time

The other major environments are connected with the setting of paragraphs. In a later chapter we will look at how LaTeX actually handles the setting of text. In the mean time we will note that sometimes it does not manage to align the right margin and complains of 'overfull boxes'. There are many solutions to this. One is to provide LaTeX with more flexibility in setting the paragraph. This can be translated as 'being more sloppy'. The two environments `sloppypar` and `fussypar` are provided to handle this. The default is `fussypar`, and therefore it is unclear just why you might want to switch it on, when a 'sloppy' set of paragraphs can be delimited easily. Note this is a paragraph matter. Since LaTeX does its typesetting on a paragraph by paragraph basis, there is no point trying to set part of a paragraph 'sloppy' and part 'fussy'. The concept just does not exist.

An alternative to this `sloppypar` environment is to use the instruction `\sloppy`. There is a corresponding `\fussy` instruction too. One of LaTeX's quirks, a consequence of treating text on a paragraph basis, is that the inclusion of one of these instructions within a paragraph, whether at the end or the beginning, turns the whole paragraph into that style; and of course every following paragraph too, unless you issue new instructions, or are careful to limit the action in some other way.

It is also possible to exercise control over the way LaTeX sets lines within paragraphs. Normally LaTeX spreads out the space between words to have both the right and left margins aligned vertically – set 'flush left' and 'flush right' (or *justified*). If, instead of adjusting the space, it is made constant, the left margin remains fixed and a ragged right margin would appear. The `flushleft` environment has this effect. Its mirror-twin, `flushright` will make the left margin ragged. And a third variant may be more useful, `center`, where the text is effectively centred around the middle axis. Taken with `\\` to break lines, this is sometimes a useful way of creating title information. All three of these environments have corresponding instructions: respectively they are `\raggedright`, `\raggedleft` and `\centering`. Again, recall that these instructions and environments will apply to whole paragraphs.

Observation will demonstrate that the `verse` environment is set ragged right, presumably to minimise hyphenation, while `quotation` is justified.

24.1 More to come

This is was just a glimpse of the environments available in LaTeX. A few more will crop up. It makes little sense to present *all* the environments at once, especially when they each have quite different functions. The key things to note are

- that they must be enclosed by the same identifier or label (e.g. `quote` or `itemize`),
- that they *may* have optional arguments,
- and that they may not overlap.

25 Sections and the like

Many (most) documents can be thought of as highly hierarchical, with a subdivision into successively smaller units, down the scale through 'parts', 'chapters', 'section', 'subsections', right down through 'paragraphs' to sentences and even words. LaTeX recognizes this and provides a set of 'sectioning' instructions, shown in Table 2Sections and the likefigure.85

Now, note that we do not in fact go down right to the sentence or word level. And also note that Lamport's `\paragraph` and `\subparagraph` are not really paragraphs at all. Lamport merely says

> "The names ... are unfortunate, since they usually denote units that are usually composed of several paragraphs; they have been retained for historical reasons."

Make of that what you will.

Each sectioning instruction requires an 'argument' – its own title. Since this is mandatory, it appears in braces. (Remember that only optional arguments appear in square brackets.)

```
\part
  \chapter
    \section
      \subsection
        \subsubsection
          \paragraph
            \subparagraph
```

Figure 2. The sectional hierarchy

3 Sections

In article and report, section is the highest level.

3.2 Subsections

After the section comes the subsection.

3.2.1 Subsubsection

After that comes the subsubsection.

Paragraph Then we have paragraphs which run on immediately after the paragraph label.

Subparagraph And at the very lowest level, the subparagraph, which, like the paragraph runs on.

Figure 3. Sub-divisions from Section to Subparagraph

What do we get out of sectioning? It is easiest to see by trying it out. As you will have realised, this document is produced with LaTeX. We have already used `\chapter` at the beginning of this 'section', using the statement

```
\chapter{Environmentally aware}
```

Somehow, LaTeX managed to work out that this is the chapter 3chapter.57, and arranged for the details to be printed out. So one of the things going on in the background is to keep track of the level of the sections. Similarly, this particular section started with a

```
\section{Sections and the like}
```

instruction. Again, LaTeX kept track of the levels. Notice too that it presented the information in different ways. There are no sub-sections here, but we could invent a few:

```
\section{Sections}
In article and report, section is the
  highest level.
\subsection{Subsections}
After the section comes the subsection.
\subsubsection{Subsubsection}
After  that comes the subsubsection.
\paragraph{Paragraph}
Then we have paragraphs  which run on
  immediately after the paragraph label.
\subparagraph{Subparagraph}
And at the very lowest level, the
  subparagraph, which, like the paragraph
  runs on.
```

yields Figure 3Sections and the likefigure.86 where the slight reduction in size is deliberately introduced to highlight these sub-divisions. Each of these sections must contain something, if only a lower level unit. At the lowest unit, there must be some text.

QUESTION 3.11 *This book makes no real use of* `\paragraph` *and* `\subparagraph` *since its author doesn't think*

in those terms. Find out how LaTeX actually handles the text which follows these sectioning instructions. The results may not be entirely anticipated.

Naturally, in order to begin (say) a subsection, you will already have begun a section. And equally naturally, you may expect to have several subdivisions within a division. On the other hand, as may be obvious here, the `book` style does not require the presence of `\part`. It can start with a `\chapter`. Similarly, the `\article` style does not use `\chapter` at all. This does mean you can string all your articles together into a book, all too easily, just by a few easy edits.

QUESTION 3.12 *Demonstrate what happens when you nest the sectioning (subsectioning, subsubsectioning. . .) instructions inconsistently.*

What else have we gained? The table of contents at the beginning is also generated automatically as a result of creating sections. Now, examine the table of contents carefully to determine which levels are actually reflected in the table of contents which is created. The section 'argument' (the bit in braces) goes into the table of contents (*toc*) in its entirety. The entry may also be used to change the running head (at the top of the page)[7].

LaTeX allows you to modify the behaviour of the sectioning instructions in two main ways:

1. we may wish a different entry to appear in the table of contents from the 'body'; or
2. we may not wish the entry to appear in the table of contents, or the running head (and by implication, not to be numbered).

Both these requirements may be fairly readily accommodated. Changing the table of contents '`toc`' entry is readily accomplished by adding an optional argument:

```
\subsection[What I really
want to appear]{Sub-sub-section}
```

This optional argument will now appear in the table of contents in place of the entry which introduces the subsection. Limiting the action, and producing a 'simple' entry is possible through the `*` form. This is another of the instructions which have a variant form, followed immediately by the `*`. In this case, it would appear as

```
\subsubsection*{An almost anonymous entry}
```

This also suppresses the increments in the counter for this particular subsection.

26 Table of contents

Having mentioned the presence of a table of contents (or a *potential* table of contents), we have to say a little more about it. This is not generated unless you do actually ask for it. How do you ask for it? Say

```
\tableofcontents
```

and LaTeX will insert the table of contents at that point. Now, if the table of contents appears somewhere at the beginning of the document, how does LaTeX 'anticipate' what the section descriptions are to be? Since it cannot know, what actually happens is that you run your LaTeX file several times. The first time round LaTeX generates a new file of the same name, but with a `.toc` extension. It is this file which is read the second time around. Note that it might still be out of step, if you have edited the file in between. Essentially what has to happen is that the very final, unchanged file has to be run *at least* twice to resolve the table of content entries (recall that it includes the page numbers too).

27 Other background activity

Although we have not touched on figures and tables (yet), there are similar instructions which allow list of figures and tables to be generated. A similar sort of process goes on. A special file is generated which contains information about the list of figures, the list of tables, or even a glossary or an index. The instructions for lists of figures and tables are:

```
\listoffigures
\listoftables
```

and the files they write are given the extensions `.lof` and `.lot`. In fact, you will find that LaTeX can be altogether profligate, creating all sorts of other files, usually with the main file's name, but different extensions. Every LaTeX job creates a `.log` and an `.aux` file. The log file can give a lot of information about what is going on. Sometimes some

[7] not here, because the *Baskerville* house style over-rides this.

of it is useful. Typically a log is produced as you run the file through LaTeX, but more information will be inserted into the `.log` file, which may be consulted later. The 'auxiliary' file is more specific and contains some useful information for subsequent LaTeX runs. It is not intended for reading by people, just LaTeX.

The truth is simply that we very rarely get everything finished, or right, the first time around – it is so tempting to run a bit through LaTeX and see if it looks any good. In passing we could demolish the myth that electronic document preparation saves any time or money, or that office productivity tools have anything to do with increasing productivity. When you can make the corrections yourself, you do, using your time, which is costly. And you seek perfection, because you know it is attainable. Before, you were just grateful to get something – anything.

Most of these extra 'information' files are there solely for LaTeX, and are of little direct use to us (except in terms of what they will allow us to do). They are just about human readable, but why bother? Perhaps the key thing to note though is that if you send off your LaTeX file to someone else, or receive one from someone else, you really should expect to have it run through your document at least twice before it looks as it should. This is a bit unexpected for the naive or inexperienced.

QUESTION 3.13 *Now have a look and see what extra files LaTeX appears to have generated for you. Look too at an old* `.log` *file to see the information which LaTeX will have issued about these extra files. As usual, it is often difficult to distinguish the useful material from the rather less useful material ('garbage'). We really need a* `.junk` *file too.*

QUESTION 3.14 *One of the things that puzzle me about LaTeX is why sections are not environments, or even lists. LaTeX will permit you to write something like*

```
\begin{itemize}
\section{...
\begin{quote}
\section{.....
\end{quote}
.
.
\end{itemize}
```

While I can see that the section may be part of an environment, I am surprised that it can span environments in this way.

Any explanations?

III TEX Live 4 — enchancement volunteers needed

Sebastian Rahtz
s.rahtz@elsevier.co.uk

The stocks of the TEX Live 3 CD-ROM, released in April, are nearly exhausted, and the time has arrived already to consider TEX Live 4. This is a public appeal for anyone interested in working on the project to come forward.

TEX Live 4 needs to improve over TEX Live 3 in the following ways, and I need people to help me in all these ways:

1. Update basic TEX family programs (notably, pdfTEX has had important changes, but there are many small fixes to look at);
2. Update font and macro packages — at least 100 new releases to CTAN in the last 4 months!
3. Install new releases of fpTEX (was web2c-win32), CMacTeX, OzTeX, emTeX and MikTeX distributions;
4. Write better install and maintenance programs, especially for Windows;
5. Write better documentation;
6. Refine and document the package cataloguing;
7. Do something (what?) to help Mac and VMS users;
8. Check the support tree for consistency and completeness.

So can you do, and do you have time for, any of the following?

- compile software on weird Unix systems?
- write install and configuration programs for Windows 9X/NT?
- classify and describe macro or font packages to the standards of Graham Williams' Catalogue?
- write new-user documentation?
- add new packages from CTAN (especially fonts)? In particular, can you produce credible and maintainable TDS trees for users working in the following areas:
  - Greek typesetting (modern and ancient)
  - Cyrillic typesetting
  - Indian languages

Many kind people helped me produce and check TEX Live 3, but there is always room for extra hands. Of course, I do not guarentee much reward, other then the satisfaction of a job well done…

It goes without saying that a fast Internet connection, constant access to e-mail, and a belief in the principles of the TEX Directory Structure, are vital qualifications!

IV UKTUG Information

The 1997–98 UKTUG committee

Chair: Philip Taylor: p.taylor@vms.rhbnc.ac.uk
Treasurer & Membership Secretary:
Peter Abbott: peter.abbott@cl.cam.ac.uk
Secretary: Jonathan Fine: j.fine@pmms.cam.ac.uk
Kaveh Bazargan: kaveh@focal.demon.co.uk
Malcolm Clark: malcolm.clark@kcl.ac.uk
Roy Everett: purpose@compuserve.com
Robin Fairbairns: Robin.Fairbairns@cl.cam.ac.uk
David Hardy: editent@btinternet.com
Sebastian Rahtz: s.rahtz@elsevier.co.uk
Kim Roberts: robertsk@oup.co.uk
Mark Wooding: mdw@ebi.ac.uk
Dominik Wujastyk: ucgadkw@ucl.ac.uk

Contacting UKTUG

Please send UKTUG subscriptions, and book or software orders, to the Treasurer: Peter Abbott, 1 Eymore Close, Selly Oak, Birmingham B29 4LB. Fax/telephone: 0121 476 2159. Email peter.abbott@tex.ac.uk.

General enquires should be sent to the Secretary: Jonathan Fine, 203 Coldhams Lane, Cambridge CB1 3HY. Telephone: 01223 215389.

UKTUG maintains pages on the World Wide Web at http://www.tex.ac.uk/UKTUG/ Email enquiries about UKTUG to uktug-enquiries@tex.ac.uk.

Baskerville

Articles may be submitted via electronic mail to baskerville@tex.ac.uk, or on MSDOS-compatible discs, to Sebastian Rahtz, Elsevier Science Ltd, The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, to whom any correspondence concerning *Baskerville* should also be addressed. Back issues from the previous 12 months may be ordered from UKTUG for £2 each; earlier issues are archived on CTAN in usergrps/uktug.

Book Discounts for UKTUG members

We have arrangements with Addison-Wesley for their well-known TEX-related publications, and with International Thomson Publishing to supply any of the very excellent O'Reilly & Associates Inc. series of books to members.

Due to price fluctuations and edition changes, it is not possible to provide an uptodate list of books. Please send details of the books in which you are interested to Peter Abbott by fax, phone or email. Peter will supply a quotation without any obligation.

We are only allowed to offer this service to **current** members of the UKTUG and/or members of TUG. Please send your order and cheque (in UK £) to Peter Abbott. Make cheques payable to 'UKTUG' please. All books will be routed through UKTUG. *In all cases* please notify Peter Abbott by email, phone, fax or letter when books are delivered. This means that provided the book(s) are in stock, it will normally take at least a week from receipt of order to delivery of the book(s).

Obtaining TEX from CTAN

The UK TEX Archive on ftp.tex.ac.uk is part of the CTAN (Comprehensive TEX Archive Network) collaborating network of archives on the Internet organised by the TEX Users Group.