ACORNSOFT
The choice of experience in software.

View

Dear Edward,

I am concerned about our poor performance during the last three months in your region.

In spite of improvement in service revenues and a recovery in new car sales after April's bad month, sales in your region are considerably below the first three months of the year. This seems to be directly attributable to a major decline in used car sales, as shown here:

	F	M	A	M	J
Total Sales	460.9	451.2	357.8	399.9	419.0
of which					
Used Cars	127.9	97.1	86.0	75.6	74.1

It seems that following our record month in February, the sales force are resting on their laurels and I would be grateful if you would let me have your proposal to correct this decline.

View

for the British Broadcasting Corporation Microcomputer

User Guide

ACORNSOFT
The choice of experience in software.

First published in 1985 by Acornsoft Limited

British Broadcasting Corporation has been abbreviated to BBC in this publication.

SECOND EDITION

Printed in the United Kingdom

ISBN 1 85250 021 2

Part no: 0449, 821

Published by Acorn Computers Limited

Contents

Introduction

What is word processing?

The easiest way to describe word processing is to compare it with using a typewriter. Think about what happens when you type a document.

First you type it in rough, so that it can be edited. You edit it, changing words, swapping paragraphs around, changing paragraph lengths, adding headings, redrafting some of it and putting other sections in tabular form.

After that you type it all again. You check it, retype parts in a narrower column width with side headings, perhaps retype whole pages where there are too many corrections.

After that you have your 'top' copy. If you want other copies, you have to photocopy, use carbons, or type it all again.

With a word processor, you type the text in as before, with the difference that the text appears on a monitor or television screen instead of on paper. The continual rapping and buzzing of conventional typewriters is replaced by the soft rattle of keys, and when you make a mistake, instead of going to a great deal of trouble to make an imperfect correction, you have only to replace one character image on the screen with another and the job is done, quickly and perfectly.

You save your rough draft on the filing system, and you print out copies on paper for checking.

You edit the draft. You find that large chunks of it are correct. With word processing there is no need ever to type these again, since they are recorded for you to use as many times as you like, in this or any other document.

You make your corrections on the screen, very easily. You insert and delete lines, move blocks of text around, and restructure the tab stops even though the text is already typed. If you want to see what the text looks like in a narrower column, you can try it out in a matter of seconds.

Then you print it out again – simply by giving an instruction to the word processor to do so.

If you want to send out several versions of your text to several companies, only changing the company name and a few other details such as the address, there are some very effective and quite simple ways of doing that too.

All the processing is done in the microcomputer and the results are displayed on the monitor, before being saved for future use, or printed out.

The VIEW word processor

The VIEW word processor is designed to do all the things described above, and much more besides.

A typical VIEW word processing system consists of the following:

– Your computer

– A good quality monochrome video monitor. Some people claim that a green screen tires the eyes less. You can also use a colour monitor or, less satisfactorily, a television set.

– A printer. What printer you use depends on your requirements. If you want the text to look as if it is typed on an electronic typewriter, you should consider a good daisy wheel printer.

To connect up your computer, monitor, disc unit or cassette recorder, and printer, see the manuals that accompany these pieces of equipment.

Before starting to use VIEW you should position the function key card at the top of the computer keyboard.

The function key card will be your guide when issuing commands to VIEW while you are processing text.

If you have used a word processor before ...

The main thing is not to jump to conclusions. Read this book with care and take it all from square one. At first you may feel that VIEW is behaving rather differently from what you have come to expect of a word processor. But if you follow the explanations and work through the examples in this book, you will soon find VIEW very simple and natural to use.

With a little experience you will quickly pass to the more sophisticated facilities. Again work through all the instructions and examples. If you skip you are sure to miss the one time-saving facility that is especially valuable for your work.

In fact experienced users of VIEW often find it valuable to look through the book every few months to remind themselves of the facilities they have not used recently and to suggest ideas for improving their work.

1 Basic techniques: typing and editing

Word processing involves typing text, saving it onto the filing-system, editing it and printing it out. This chapter and the four that follow concentrate on the basic methods of doing all this with VIEW.

So by the time you have read and worked through chapter five, you will be in a position to carry out straightforward word processing.

From chapter six onwards the book describes extensions and refinements of the basic methods. These more advanced methods will make your routine work easier and quicker, and will greatly extend the range of things you can do with VIEW.

Using VIEW

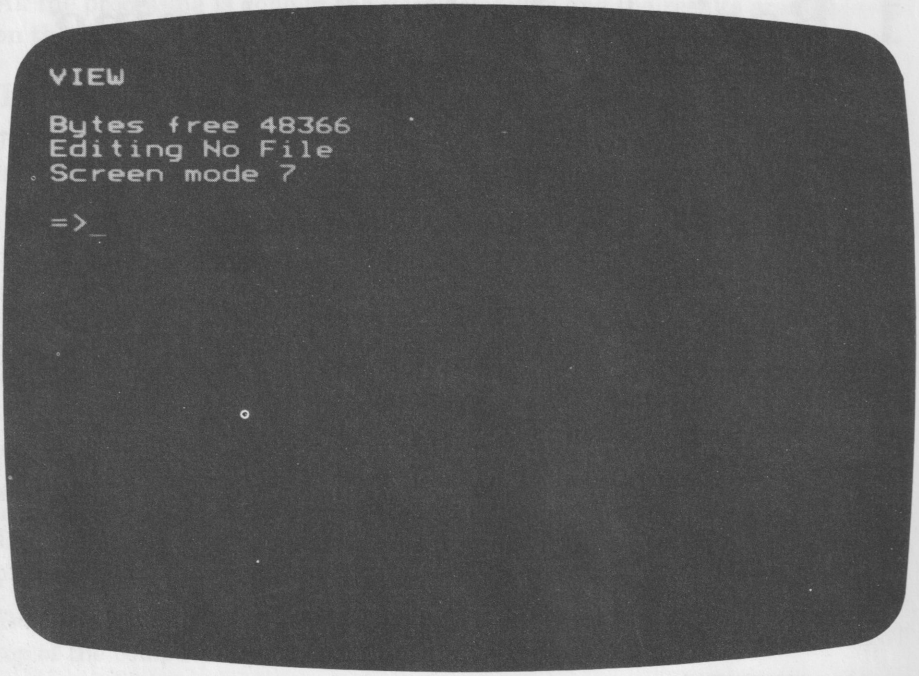
To get into VIEW type:

`*WORD` **RETURN**

That is: type `*WORD` and press **RETURN**

If you are loading VIEW from disc, first insert the disc into your disc drive.

Feel free to experiment with any keys – you cannot do any harm by pressing them.



Commands

In this guide, commands are shown in the following forms:

`Computer typeface`

Type in the command exactly as shown. Commands may be typed in either upper or lower case though in this manual they are shown in upper case.

`Italic computer typeface`

Type the appropriate name or number.

Bold upper case

Press the key specified.

Examples

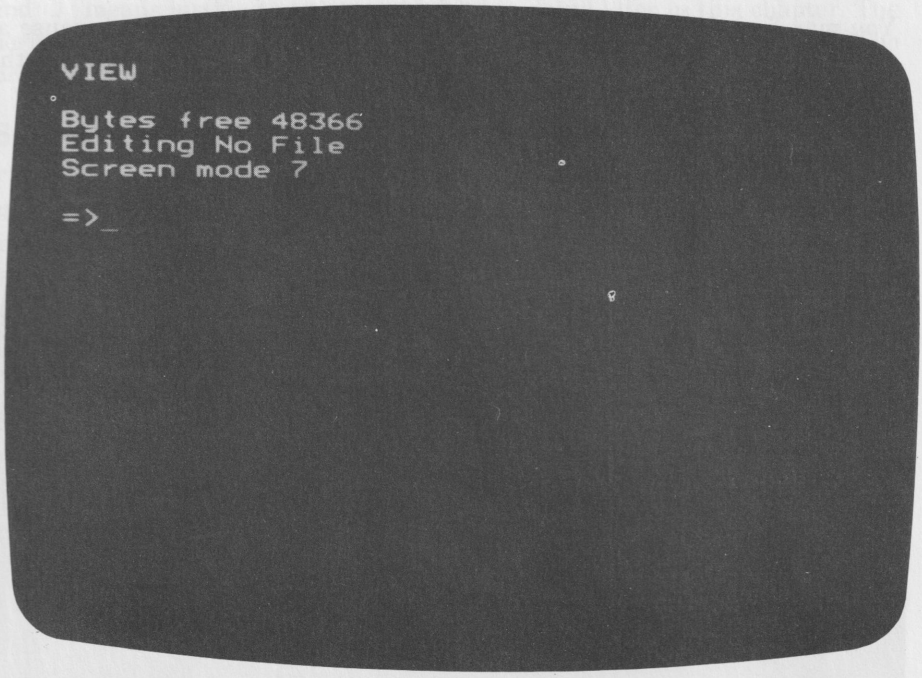
`LOAD` *`filename`* **RETURN**

means type the word LOAD exactly as shown; type a filename of your choice; press the RETURN key.

`MODE` *`number`* **RETURN**

means type the word MODE; type a number; press RETURN.

The command screen



The top line of the display confirms that you are in VIEW.

The phrase `Bytes free 48366` (or some other number, depending on how your computer is set up) shows the amount of memory free for you to use. As a rough guide one byte corresponds to one character – ie one letter, number, sign or space.

The phrase `Editing No File` shows that you have not loaded a text file; if you do so this will change to `Editing` *`filename`*.

`Screen mode 7` is a reminder of which screen mode you are in.

The sign `=>` shows where commands are typed in.

VIEW uses two types of screen

– The command screen in which you issue general commands to the system.

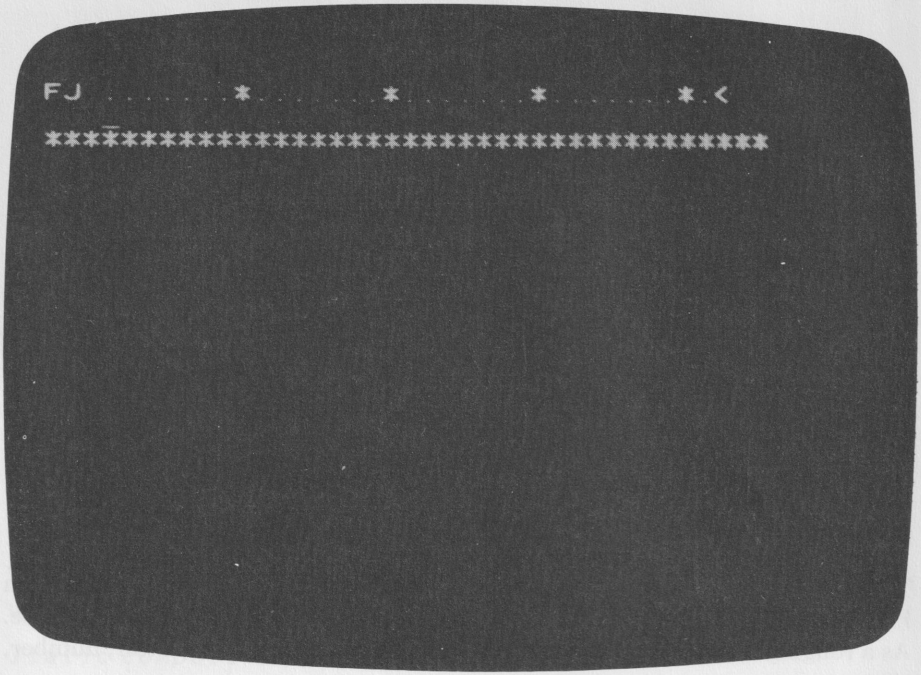
– The text screen in which you write and edit text.

You can switch at will between the two screens by pressing **ESCAPE**.

Try switching to the text screen and typing in some text. When you want to move down a line, press **RETURN**.

You may find that you are typing in capitals all the time. This is because at start-up the 'caps lock' function is on, as signalled by the red light on the keyboard. To switch it off press the **CAPS LOCK** key.

The text screen

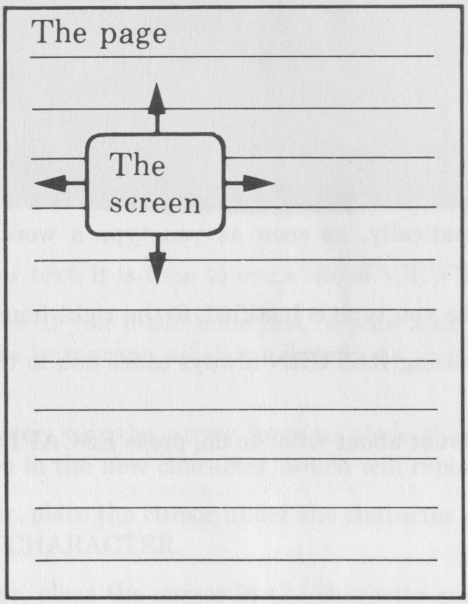


This is screen mode 7, capable of displaying 24 lines of 34 characters each in VIEW.

The top line is the ruler, and this controls the width of the column of text under it. The asterisks in the ruler are tab stops. The letter F means format, and J means justify; these are explained in detail later in this chapter. The horizontal bar is the cursor, and shows where any new text will be typed. The row of asterisks marks the bottom of the column of text.

Screen and page

The four arrow keys move the cursor about the screen. If you use them at this stage it is important to realise what they are doing. VIEW's text area is not limited to the screen itself. The best way is to regard the text area as a very large 'page', only a little of which is visible to you through the screen. The screen in fact is rather like a window which you can move (using the arrow keys) to any part of the page you wish.



Immediate commands

When you are in the text screen, you can use the commands on the function keys. These are called immediate commands because you do not have to switch to the command screen to use them but can use them as you are editing your text. Look at the function key card, which you should keep inserted above the function keys at the top of the keyboard. The most frequently needed commands are on the bottom row, and these are used by pressing the function key above them. The commands on the second and third rows are used by

pressing the function key at the same time as you press **SHIFT** and **CTRL** respectively.

The instructions in this book show the immediate commands differently from the command screen commands described above. Here the key function is shown in ordinary upper case letters. So, for example, you may be told to press the FORMAT PARAGRAPH function key.

For the position of the function key refer to the function key card.

Typing text

At this stage the best way to understand VIEW is to type in any text you like, not concerning yourself in the least whether it is right or wrong.

To show something of how VIEW works type in the paragraph shown on the next page without ever pressing **RETURN**. The example assumes you are in mode 7; if you are not, return to the command screen by pressing **ESCAPE**, type:

`MODE 7` **RETURN**

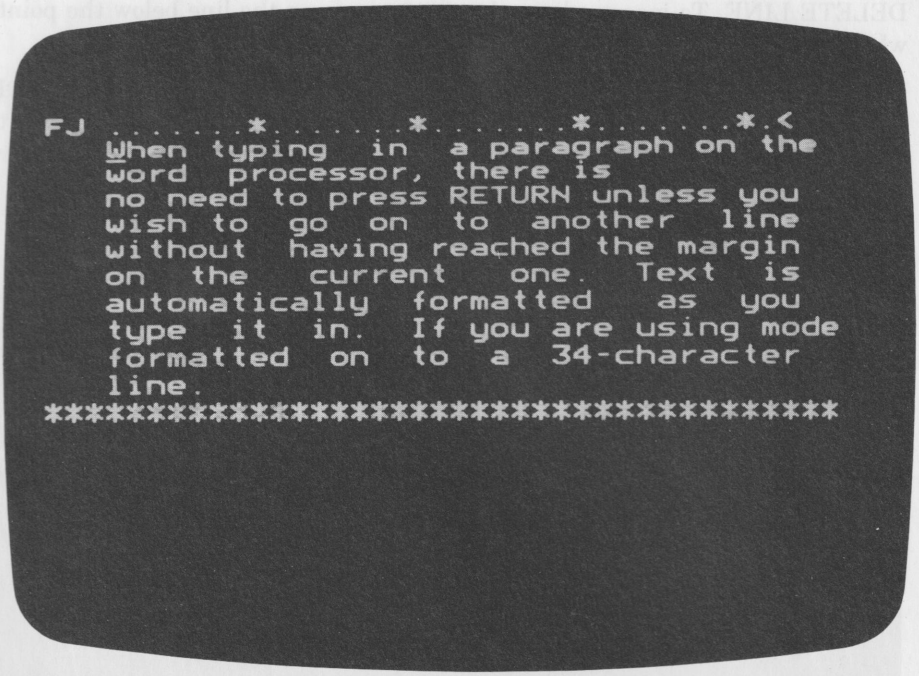
and go back to the text screen by pressing **ESCAPE** again.

You do not need to press **RETURN** as you type because VIEW moves you on to another line automatically, as soon as you type a word beyond the right margin.

Notice how each line you type is justified, ie the right-hand margin is even.

Remember that pressing **RETURN** always takes you to the beginning of the next line.

If you are ever in doubt about what to do, press **ESCAPE**.



Having typed in your text, it is time to use some of VIEW's editing facilities.

Correct any mistakes. If you made none just rewrite a bit instead, or if you prefer try to edit the paragraph as shown below. As you do so, try out the following commands.

To replace a character, use the arrow keys to place the cursor under the character. Then type in the new character, which will replace the old one.

To delete a character, place the cursor under the character you want to delete and press DELETE CHARACTER.

To insert a character, place the cursor in the character space to the right of where you want the new character to be, like this:

```
WORD PROESSING
```

Then press INSERT CHARACTER. The text will open up like this:

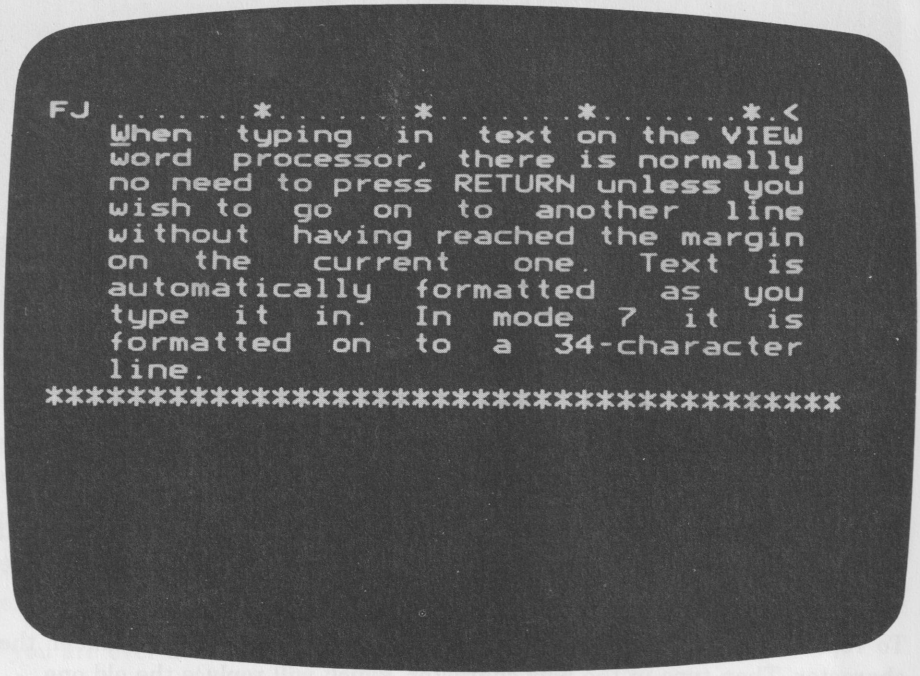
```
WORD PRO_ESSING
```

and you can type the new character.

Deleting and inserting lines is done in much the same way as deleting and inserting characters. To delete a line, place the cursor on it and press

DELETE LINE. To insert a line, place the cursor on the line below the point where you want to insert a line and press INSERT LINE.

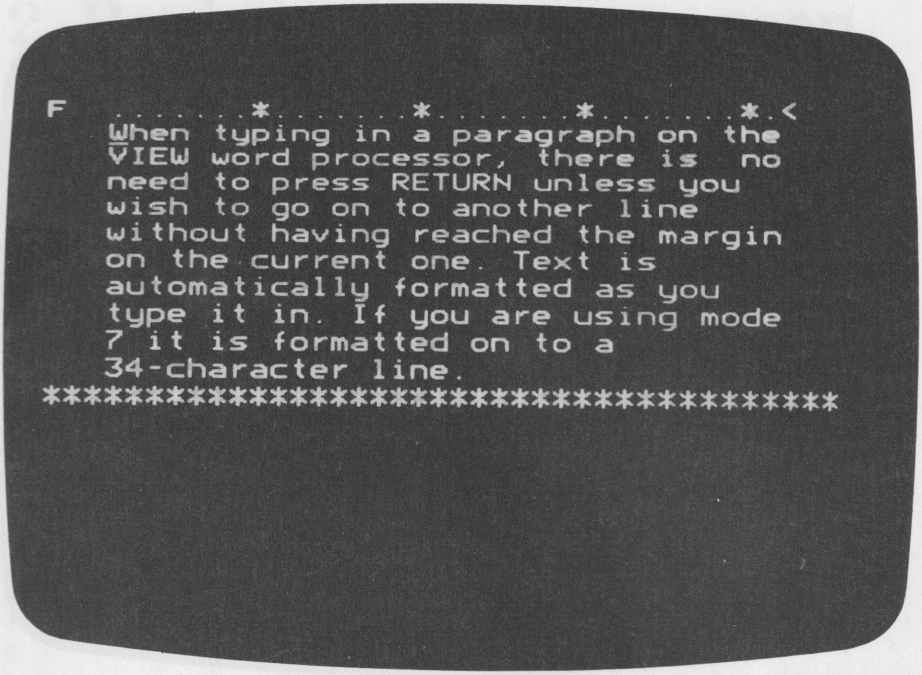
You will find that as you insert spaces the text disappears off the right-hand edge of the screen. Don't worry! It is not lost, merely extended over into a part of the page which is not covered by the screen. You can, of course, look at it by using the arrow keys to move the cursor to the right.



You can tidy up a column of text like this by reformatting it. We shall go into formatting in detail later on. The text is rearranged so that instead of having short and long lines as shown in the screen above, it all fits into a neat column.

To format the text, place the cursor anywhere on the top line of the text and press FORMAT PARAGRAPH. Make sure you do not have 'shift lock' on or you will get a bleep and no action. (If the 'shift lock' light is on, press the **SHIFT LOCK** key to turn it off.)

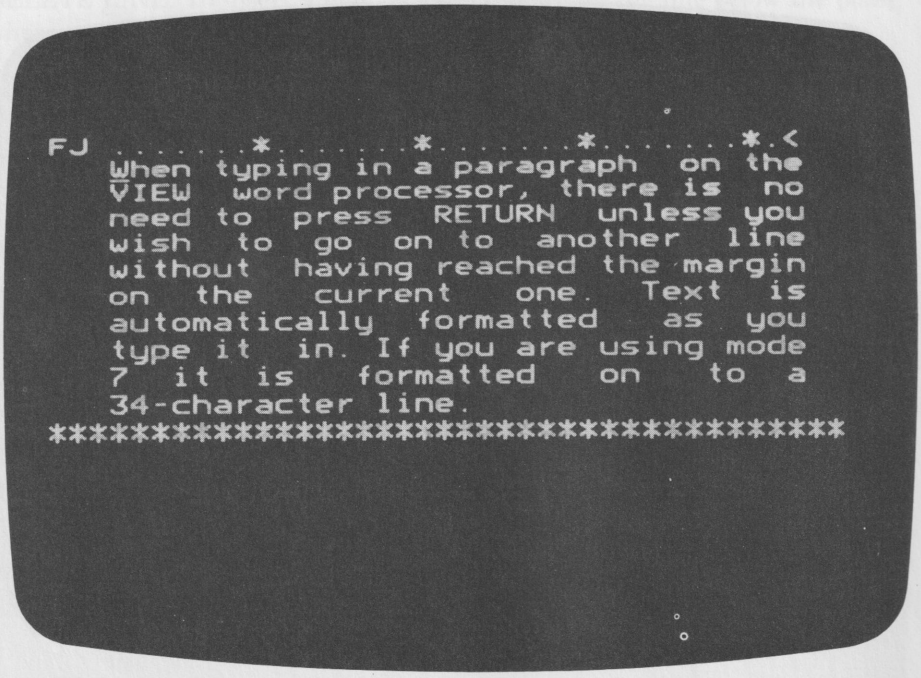
The whole paragraph is then changed into something like this.



This is what is known as 'justified text', ie all the lines are of the same length, like a newspaper column. If you look at the spaces between the words you will see that these are adjusted to achieve this effect. VIEW can make these adjustments automatically as you type text in, or afterwards when you format a paragraph as above.

Of course not everyone likes justified text, and for some purposes it looks too formal. To switch off justification, press JUSTIFICATION. The letter J to the left of the ruler at the top of the screen disappears – we are no longer justifying text – but the letter F remains, since we are still formatting.

To rearrange the text in unjustified lines, just carry out the same procedure as before: place the cursor on the top line of the paragraph and press FORMAT PARAGRAPH. The result should be something like this.



Of course, if you turn off justification before you start typing, the text you type will be unjustified. To turn justification back on, press JUSTIFICATION again.

Clearing text

When you have finished experimenting with this text, you can clear it from text memory by switching to the command screen and typing:

NEW **RETURN**

2 Basic techniques: screen modes and the ruler

Screen modes affect the appearance of text on the screen. There are eight to choose from and VIEW will work in any of them. In practice most people use mode 3 which has an 80-character line. Since VIEW offers a line 6 characters less than the current screen mode's line, this gives a 74-character line which is very suitable for laying out typewritten material. VIEW offers the following characters and lines for the specified screen modes.

Mode	*Characters*	*Lines*
0	74	31
1	34	31
2	16	31
3	74	24
4	34	31
5	16	31
6	34	24
7	34	24

To switch to another screen mode, enter the command screen and type

`MODE` *`number`* **RETURN**

For example: `MODE 3` **RETURN**

If your computer has shadow memory available, you are recommended to use it by typing:

`*SHADOW` **RETURN**

before selecting a mode.

Modes and memory

Different modes take up different amounts of memory. If you look at the heading in the command screen you will see the amount of free memory shown in the form `Bytes free ...`

This shows the amount of memory left for you to use after the text already there is taken into account. A byte is a unit of memory, equivalent to a single character on the screen – and remember that spaces, numbers and punctuation marks are also characters. As a rough guide, a typical A4 page takes about 2000 bytes.

When you start in mode 7 with VIEW in a BBC Microcomputer Model B you have about 25000 bytes of memory for your text, whereas mode 3 only allows you about 10000, and in mode 0 you are down to about 6000 before you start.

All these numbers are modified if you have 'shadow memory', which gives you the same amount of memory in all modes.

The ruler

You have probably noticed the row of dots and asterisks at the top of the text screen. This is the ruler and it controls the length of the lines typed beneath it.

Rulers can be more clearly explained if you switch to mode 3 at this point. This is the mode in which you are most likely to do all your typing and editing.

Clear any text you have typed in by switching to the command screen and typing `NEW` **RETURN**.

Type `MODE 3` **RETURN**

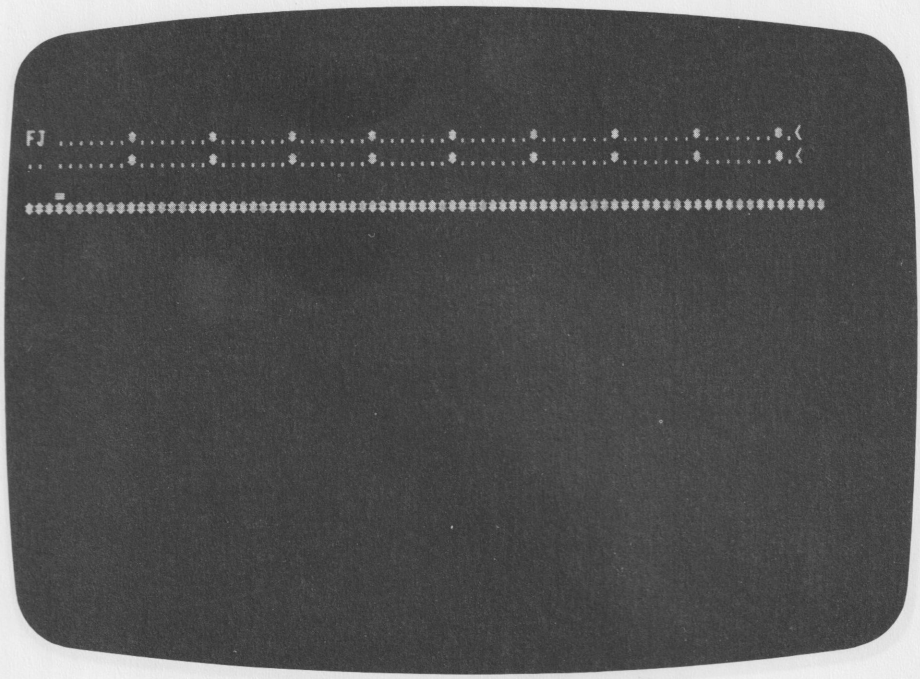
Type `NEW` **RETURN**

Press **ESCAPE**

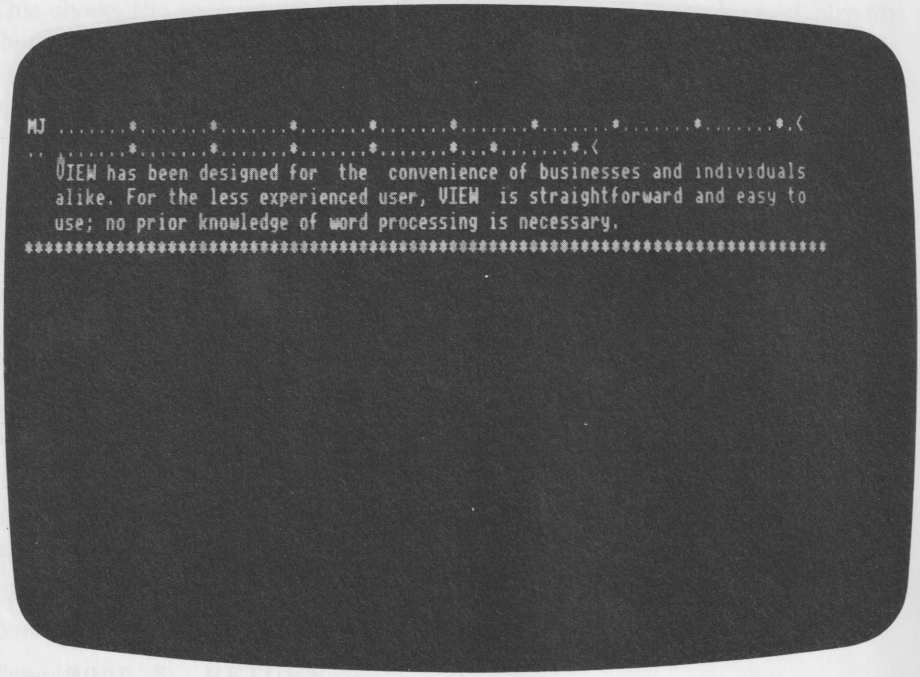
Press RULER

Press **RETURN**

You are now in mode 3 with the standard ruler as the current ruler.

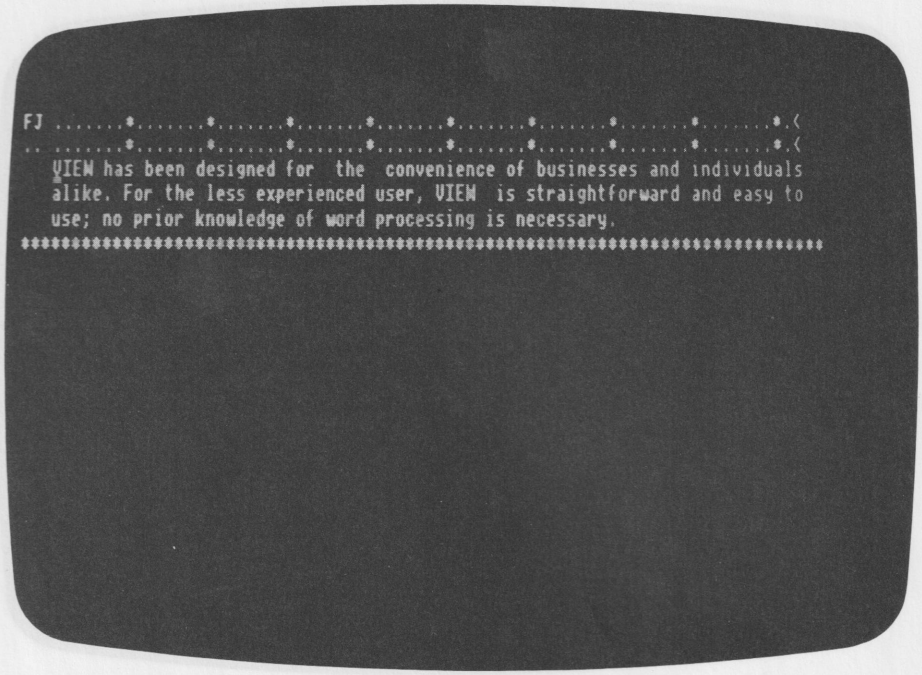


The current ruler is the one which is above the cursor position.The current ruler controls the length of the lines typed beneath it. Any text you now type in will be limited to the length of the ruler. Type in the text shown opposite.



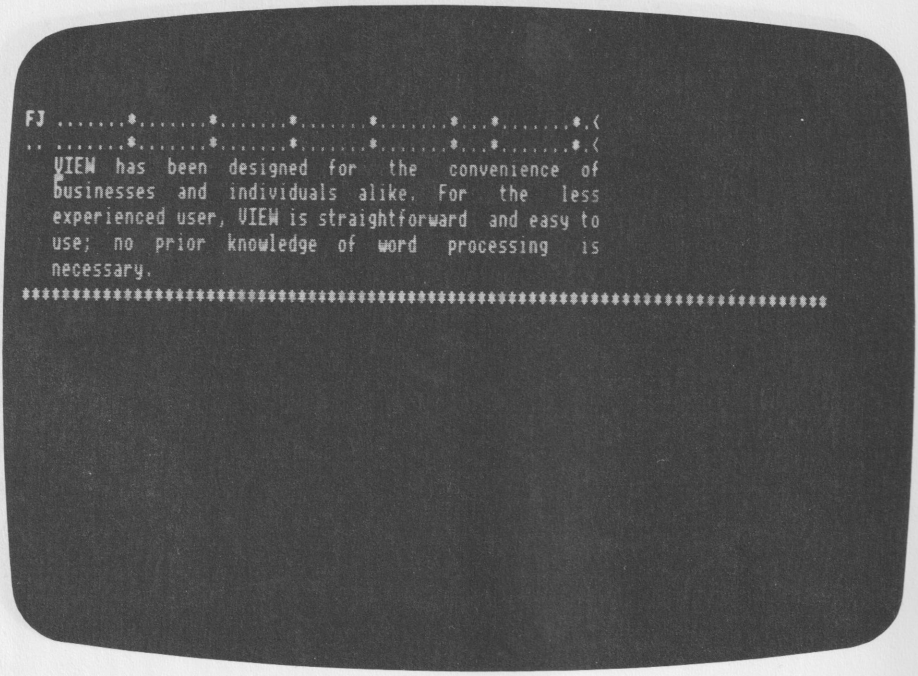
Editing the ruler

If you want to change the layout of your text, you will have to alter the length of the current ruler. Move the cursor back onto the current ruler and change it so that it looks something like the example shown overleaf. You can use the same editing facilities as you have been using for editing text, DELETE CHARACTER for example.

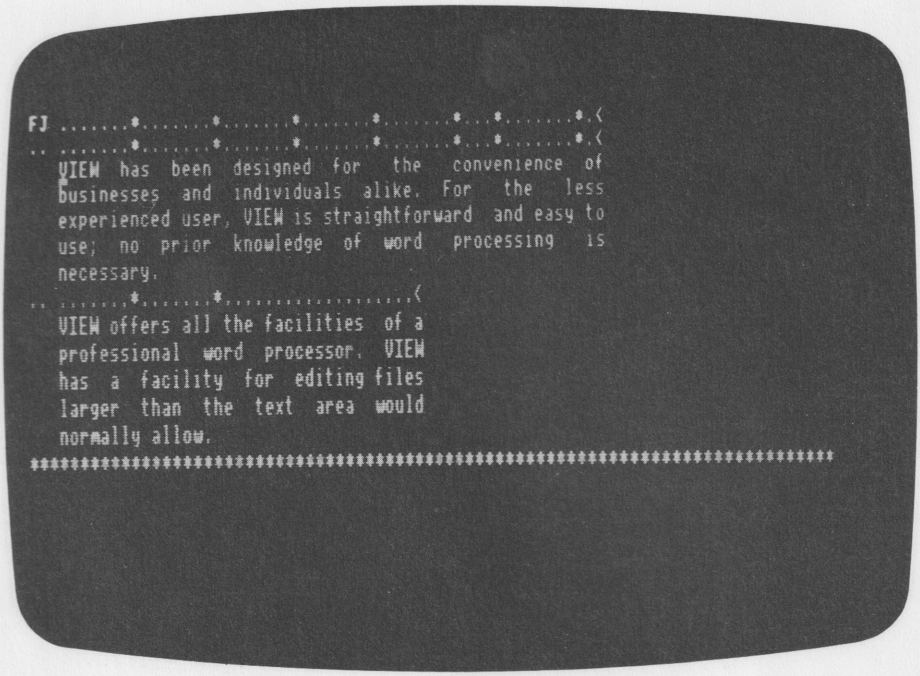


Note that when the cursor is on a ruler, the `F` flag changes to `M`. This is because, when the cursor is on a ruler, margins are inactive.

Now move the cursor off the newly edited ruler by pressing **RETURN**. Press FORMAT PARAGRAPH. The text will now align itself with the new current ruler as shown opposite.



Notice that the ruler at the top of the text screen has now changed to match the new current ruler. If you change the current ruler, the ruler at the top of the screen will also change. This can be a useful reminder of which ruler is active at the cursor position.



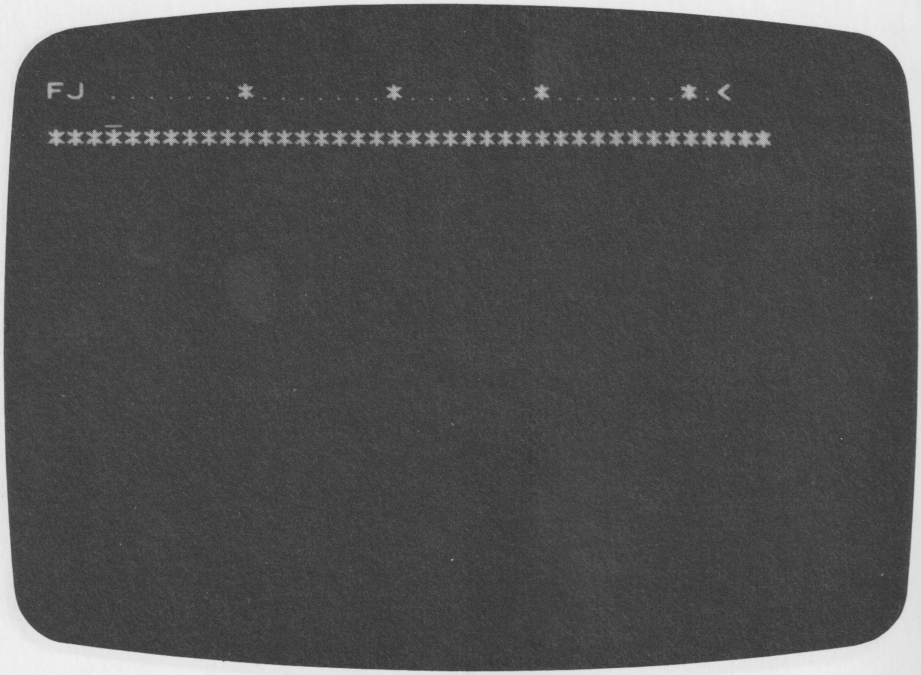
In the above example, a ruler has been added in the middle of the document and more text has been typed in. There are two ways to do this:

– Put a standard ruler on to the screen by pressing RULER. Each mode has its own standard ruler. Pressing RULER puts the standard ruler for the current screen mode on to the screen. You can then edit this ruler as you wish. It is a good idea always to put a standard ruler at the very start of your document too.

– Copy the current ruler by pressing **SHIFT** and **COPY** simultaneously. You can then edit this ruler as you wish.

Starting up

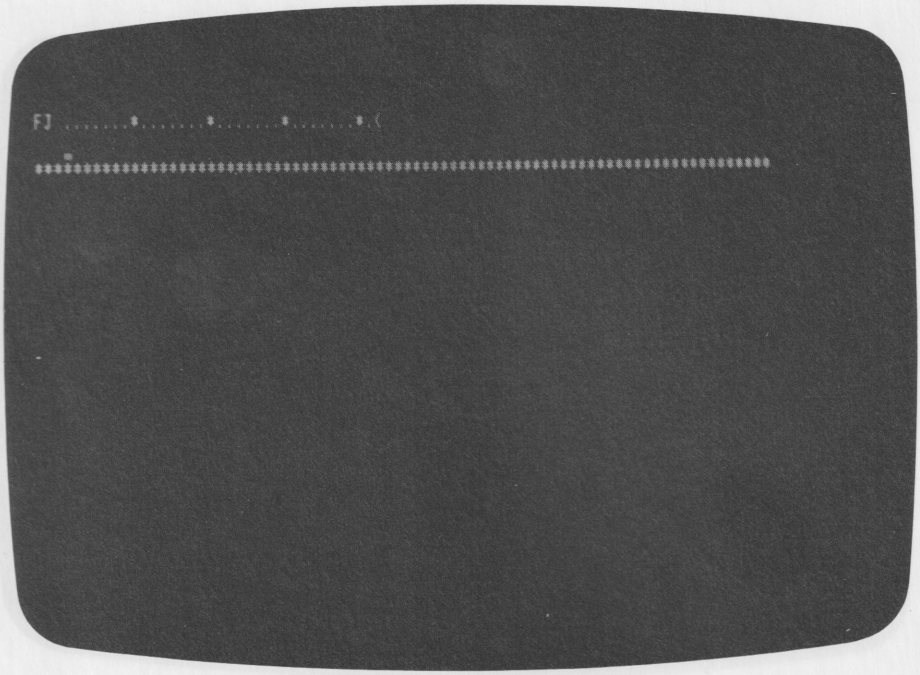
If you start up with VIEW in mode 7 you will see the following when you switch to the text screen.



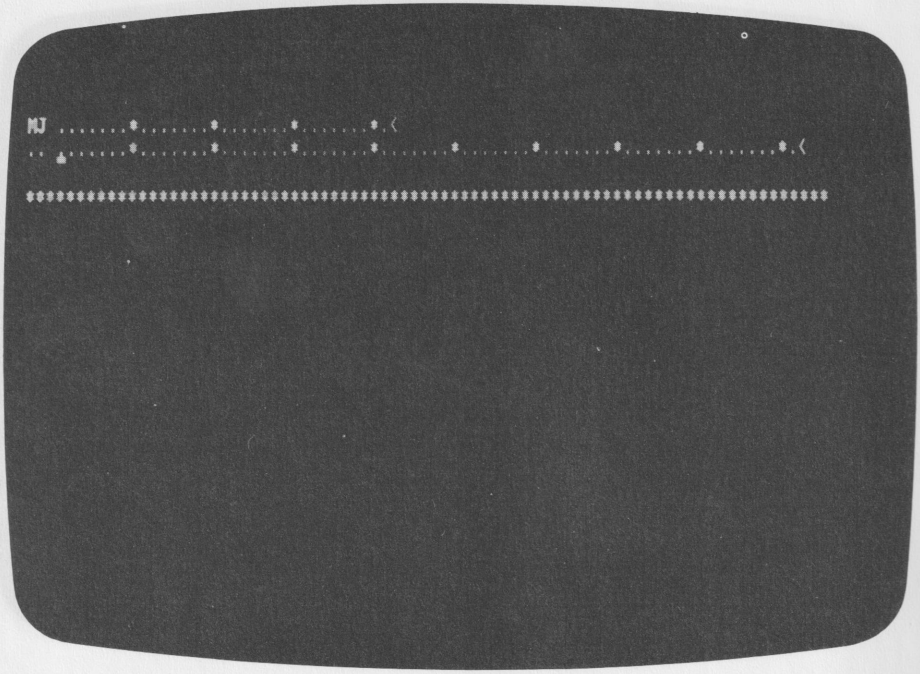
If you switch to the command screen and type

MODE 3 **RETURN**

and then switch to the text screen, you will see



Although you are now in mode 3 you still have the mode 7 standard ruler on the screen. Press RULER to get the mode 3 standard ruler.



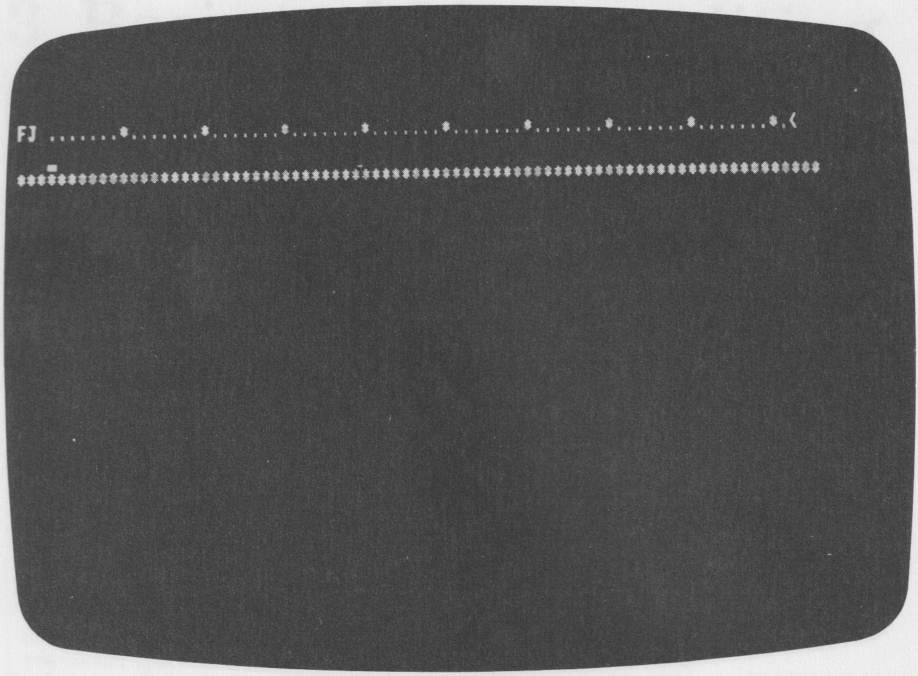
Alternatively, on entering VIEW, you can type

`MODE 3` **RETURN**

then type `NEW` **RETURN**

and press **ESCAPE**

which will produce the following on the text screen.



It is still a good idea to press RULER at this point to be sure that if you are editing the same text at a later date it is not mistakenly reformatted under a different ruler.

How the ruler is made up

Conventionally, the VIEW ruler is made up of:

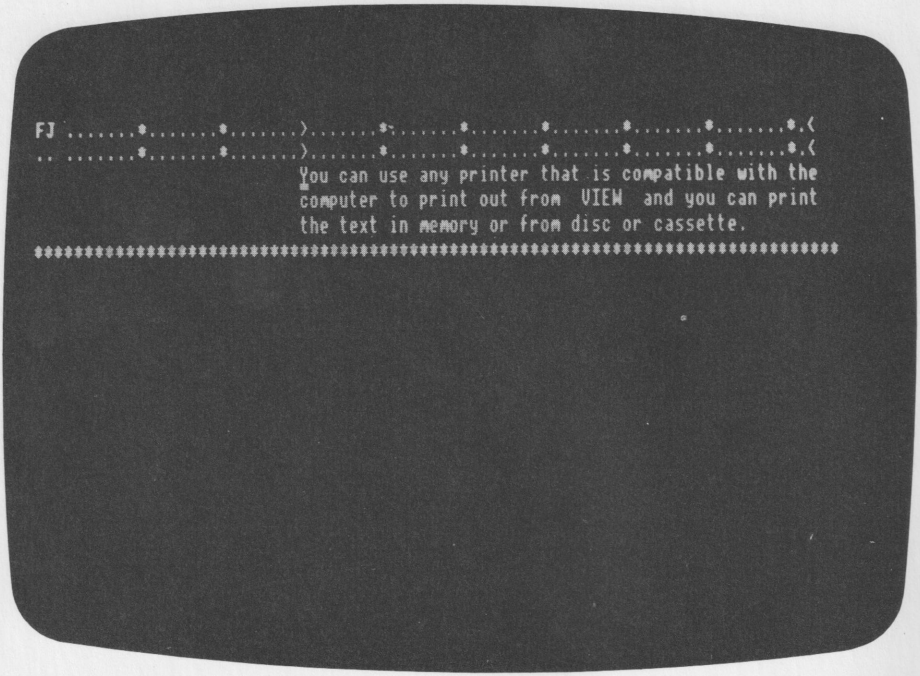
– Dots, which are there to remind you that this is a ruler.

– Asterisks, which are tab stops.

– The < character, which is the right-hand margin stop.

We shall deal with tab stops more in a later chapter. For the moment, you can use them by pressing the **TAB** key, so that the cursor jumps along the line to a point level with the next stop.

You may also use a left margin stop on the ruler, like this:



When typing text, the cursor lines up with the left margin stop and the first character of the line appears at this point.

You may also insert the character b or B in a ruler. This causes the computer to bleep when the cursor reaches this point, just like the bell on a typewriter, except that VIEW allows you to place it wherever you like, and it has no connection with margins.

3 Basic techniques: editing text

Now for some editing. One of the best things about word processing is its facility for invisible mending. Absolutely anything can be changed – words replaced, whole paragraphs inserted or deleted, or moved from one place to another – and when the job is done it is as if the text had never been otherwise.

In the example below the author typed the main explanation first, followed by other material which might appear before or after it.

```
PEBBLE GRADING BY WAVE ACTION

The swash of a wave is more powerful than
the backwash. This is because the swash
loses a lot of its volume in the shingle
before it starts to return, and becuase
the swash starts with the impetus of the
breaking wave, whereas the backwash does
not.

Many people are astonished to find that on
most beaches they visit the biggest
pebbles are at the top of the beach, the
smallest at the bottom, and in between
those of intermediate size.

The effect is to produce a grading of
pebbles over the beach from the largest at
the top down to the small pebbles that
look like a bed of gravel before the sand
that extends under the waves.

Why should the largest pebbles be carried
furthest? The explanation is quite simple.
```

Try typing it all in yourself, and carry out the editing process that follows.

Having typed it all in the author prints it out. He decides that although he has made most of the points he wanted to make, some of it could be better expressed, and a few words he has used should be explained. So he edits the text like this.

PEBBLE GRADING BY WAVE ACTION

The swash of a wave is more powerful than the backwash. This is because the swash loses a lot of its volume in the shingle before it starts to return, and because the swash starts with the impetus of the breaking wave, whereas the backwash ~~does not~~ starts with no impetus.

Many people are ~~astonished~~ surprised to find that on most beaches they visit the biggest pebbles are at the top of the beach, and the smallest at the bottom, ~~and in between those of intermediate size~~.

The effect is to produce a grading of pebbles over the beach from the largest at the top down to the small~~pebbles~~est, that look like a bed of gravel before the sand ~~that extends under the wave~~.

Why should the largest pebbles be carried furthest? The explanation is quite simple.

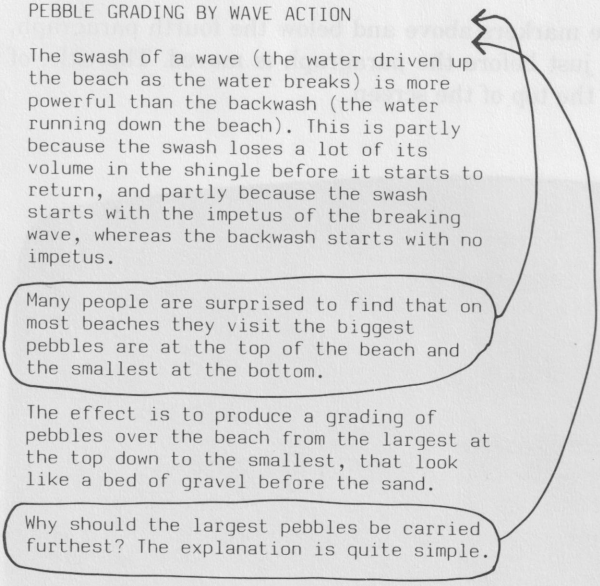
We have already mentioned INSERT CHARACTER and DELETE CHARACTER. Clearly all the amendments specified above can be made using these two facilities alone – and if you feel like having a little practice with them, go ahead and use them.

But some of the insertions are rather long. The quickest and easiest way to do these is to use INSERT/OVERTYPE. When you press this the letter I appears at top left on the screen, alongside the F and J . This is to remind you that you are inserting. If you now position the cursor (using the arrow keys) where you want to insert text, and start typing, the text already there will open up and the words you are typing will be inserted. To tidy up the words you are typing and correct any mistakes, you can still use INSERT CHARACTER and DELETE CHARACTER as usual.

Depending on how you go about the task, you may find DELETE LINE and INSERT LINE useful too.

When you insert text you will find that the lines take on a very ragged appearance at the right margin, but when you re-format them, all the words return to the confines of the ruler again. Remember that formatting is done by placing the cursor on the top of the paragraph and pressing FORMAT PARAGRAPH.

When all the corrections are done and the text is reformatted, the result should be like this. But already the author has marked new corrections. He finds that he has given an explanation in his text before actually posing the problem, so he wishes to move two of the paragraphs to new positions as shown below.



So paragraph two becomes paragraph one, and paragraph four becomes paragraph two.

Moving blocks of text in VIEW is done by setting 'markers' before and after them and then telling VIEW to move the text which lies between the markers.

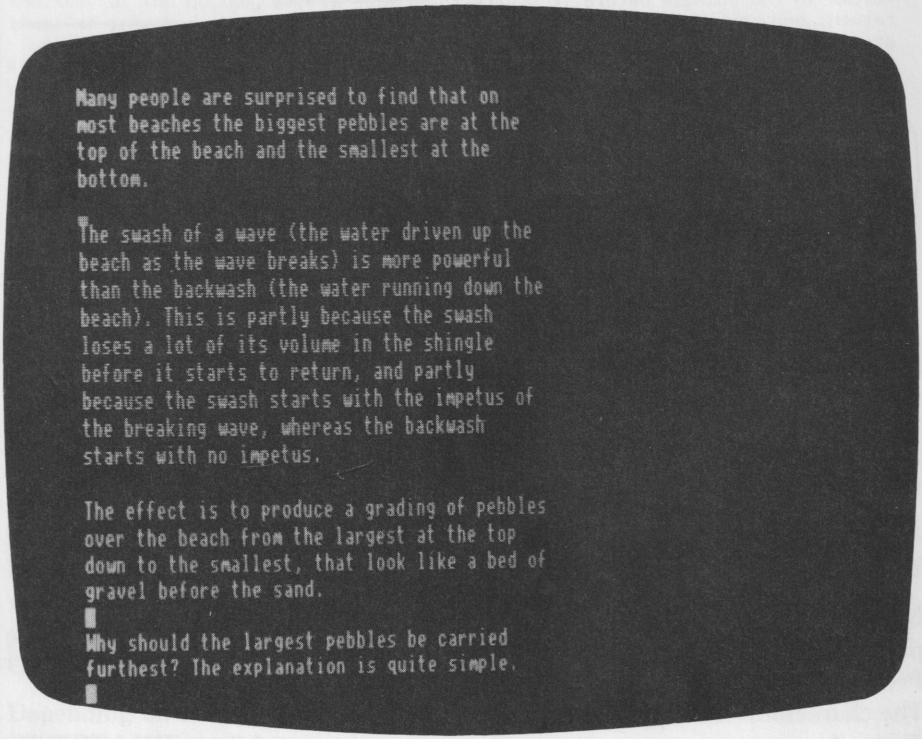
Move the cursor to the point immediately above the first letter (`M`) of the second paragraph ie on the line between the first two paragraphs.
Press: SET MARKER.
The characters `MK` appear at top left.
Type: `1`
A white block appears to show the position of marker 1.

Move the cursor to the line below the last line of the paragraph.
Press: SET MARKER.
The characters `MK` appear at top left.
Type: 2
A white block appears to show the position of marker 2.

Move the cursor to the line below the title.
Press: MOVE BLOCK
Paragraph two becomes paragraph one.

Carry out the same procedure with paragraph four, moving it to the paragraph two position.

The screen below shows the markers above and below the fourth paragraph, and the cursor in position, just before the paragraph is moved. The title, of course, has disappeared off the top of the screen.



We have gone into moving text at some length, partly because it is such a useful method, and partly because much the same procedure is used to delete blocks of text, and to copy them.

To copy a block of text, mark it as if you were going to move it. Then move the cursor to the position where you want the copy, and press the **COPY** key.

To delete a block of text, mark it in the same way, and press DELETE BLOCK.

You may have noticed that when you have moved or deleted a block of text, the markers disappear. When you copy a block of text, the markers remain on each side of the original block, so that you can copy it many times if you wish, just by moving the cursor to the place where you want the copy and pressing **COPY**.

4 Basic techniques: saving and loading

We said earlier that a valuable feature of word processing is that you can save text permanently. When you want to revise it you can load it into the computer and work on it as often as you wish.

We shall go into saving and loading in more detail later in this book. The basic techniques are quite straightforward.

Saving text

Saving and loading are done in the command screen. To save the complete contents of the text memory (as shown on the text screen), switch to the command screen and type:

`SAVE` *`filename`* **RETURN**

There are limits on the length of filenames, depending on which filing system you are using. For the moment you can avoid any problems by restricting your filenames to seven characters with no spaces.

If there is another file of the same name on the filing system, this command wipes it out and replaces it with the new file – which can be very useful if you are continually improving your text and saving the latest version each time you work on it. It can also be very frustrating if by mistake you overwrite a file you wanted to keep.

So always keep back-up copies of the texts you cannot afford to lose.

Loading files

To load a file, switch to the command screen and type:

`LOAD` *`filename`* **RETURN**

If a VIEW file of that name exists it will be loaded. Any text that was on the text screen at the time will be replaced by the new text from the file.

At the same time the command screen message: `Editing No File` will change to: `Editing` *`filename`*. The `Bytes free` message will also change to indicate how much memory is still free after the file has been loaded.

If the file you have asked for is not found (or if you have mis-spelt its name, which to the computer is the same thing!), you will see the message: `File not found`.

If you are trying to load a VIEW file, but accidentally name a file of another kind (eg BASIC) instead, the file may appear to load, but when you try to switch to the text screen VIEW will not do so. Instead you will see the message: `No Text`. You can only switch to the text screen if you first type

`NEW` **RETURN**

You can, of course, go on to load the right file immediately.

Adding text to text

We mentioned above that when you `LOAD` a file, the text currently in the computer's memory is wiped. Sometimes, however, you may wish to preserve that text and add to it the contents of another file, so as to combine the two into a single text.

You can do this by using the command `READ` instead of `LOAD`, naming the file in the same way. If there is not enough memory to contain the whole of both texts, VIEW will read as much of the new text as possible adding it to the end of the text in memory. The amount of free memory shown in the command screen will be adjusted once the new text is read in. If you want the new text to be added to a specific place in the text in memory, rather than added to the end, then set a marker 1 at the appropriate place and type, in the command screen

`READ` *`filename`* `1`

Quick SAVE

The instructions above for saving files are to use the command `SAVE` followed by the filename. This is not always necessary.

If a file is loaded, its name appears in the `Editing` line on the command screen. If you then edit the text and want to save it with the same name as before, you have only to type:

`SAVE` **RETURN**

You can also change the name given in the `Editing` line, or introduce a name into it. This is often convenient when you have read in text, as described above, and wish to rename a combined file, or when you have started a new document. To change the filename in the `Editing` line, type:

`NAME` *`filename`* **RETURN**

Names of files

One final point: how should you name your files?

Remember to beware of the use of spaces in filenames. For example, you might type the first chapter of a report and save it under the name `ABC`; you might then go on to type the next chapter and save it in a second file which, quite naturally, you call `ABC 2`.

Unfortunately the computer takes the space between the `C` and the `2` to indicate that this is the end of the filename. To the computer, therefore, you have saved one file called `ABC` and then another file also called `ABC`, which overwrites the first. The cure for this is to miss out the space and call the second file `ABC2`.

Think out your filenames carefully. Ideally they should be unique and if some of them relate to each other (like chapters in a book) they should reflect this fact. It is also best if they actually mean something and are not just random sequences of letters which you will have difficulty in remembering. It is very boring to have to load and examine all the files in the filing system because you cannot remember what the filenames mean.

For more information on filenames see the Reference chapter at the end of this book.

5 Basic techniques: printing

Stored commands to control printing

Our final chapter on basic techniques concerns printing, but there is more to printing than simply typing in a command to get the printer working. VIEW offers many ways in which you can control and enhance the presentation of your text.

A whole set of 'stored commands' are available for this purpose. They are entered in the stored command margin to the left of the text area on the text screen, but have no effect on the text as shown there at all.

Only when the text is printed do they come into play, to control the centring of titles, page headings, page numbers, page length, and so on.

Take the centring of titles, for example. In conventional typing this is a matter of laboriously counting spaces. Using VIEW editing such as we have discussed up to now it would be easier – a matter of inserting and deleting character spaces. With a stored command it is even simpler.

Try this. First clear all the text from VIEW by switching to the command screen and typing:

`NEW` **RETURN**

Switch to the text screen.

Now type the heading we used in an example in an earlier chapter:

`PEBBLE GRADING BY WAVE ACTION`

Leave the cursor on the line (or move it back if you have pressed **RETURN**).
Press EDIT COMMAND
The cursor moves into the stored command margin.
Type: `CE` **RETURN**
The cursor moves back into the text area and the characters `CE` remain in the stored command margin.

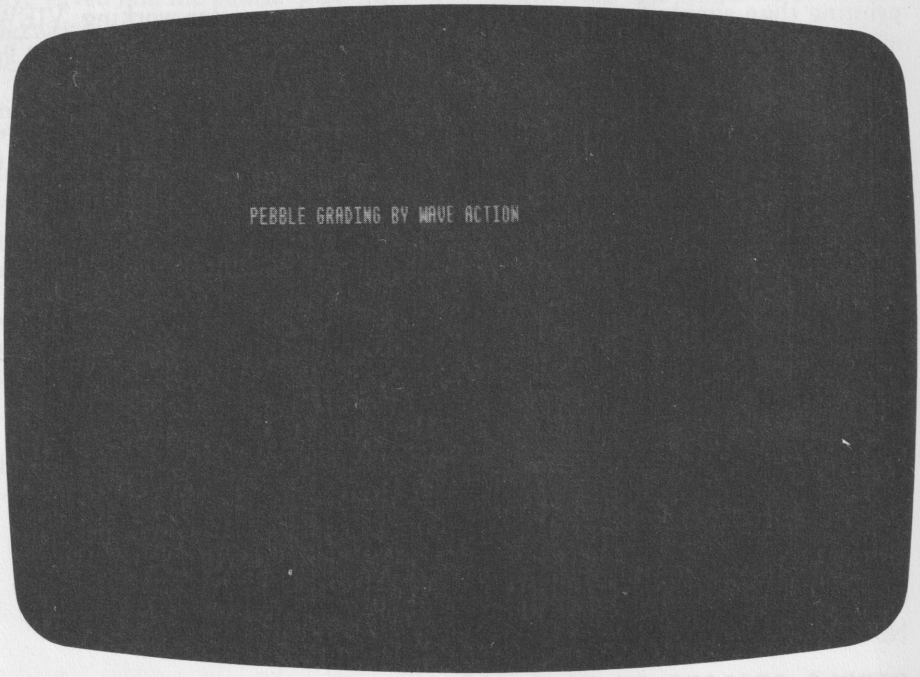
You have just entered a stored command. If you made a mistake and entered the wrong characters, you can delete them by using DELETE COMMAND.

To see the effect of the command immediately, without using a printer, you can send a simulated print-out to the screen. Switch to the command screen and type:

`SCREEN` **RETURN**

Press **SHIFT** until the prompt returns.
Press **ESCAPE** to return to the text screen.

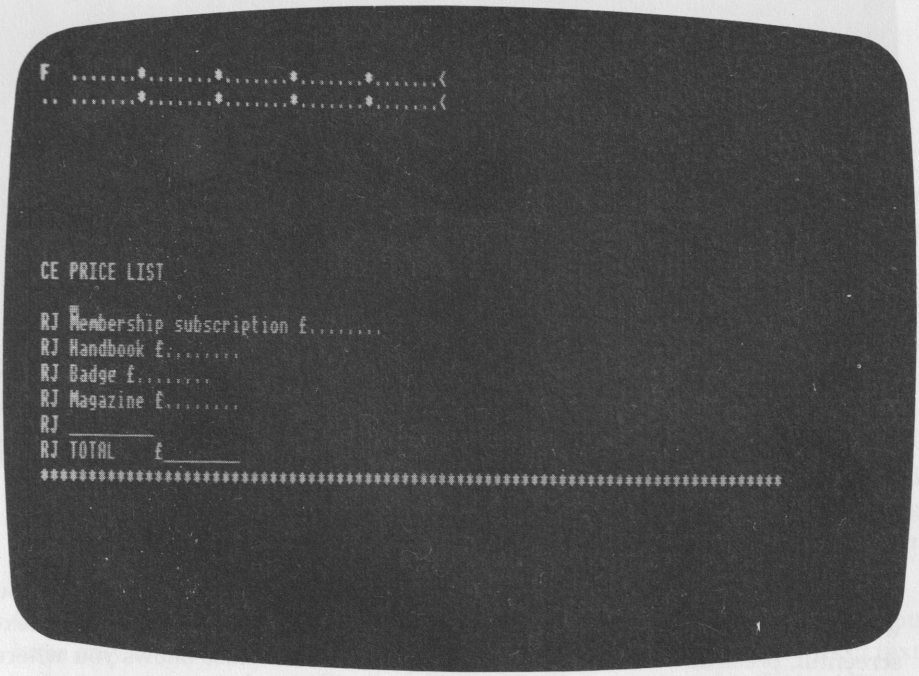
What you should have seen is the title centred on the screen, like this.



The stored command `CE` centres the text beside it in relation to the current ruler.

The `SCREEN` command is a quick way of checking on the appearance of text when printed. If your text lines are long the display is less realistic since they 'wrap around' on the screen, but with lines up to the length of the standard ruler in mode 3 the effect can be judged quite well, including the action of the stored commands.

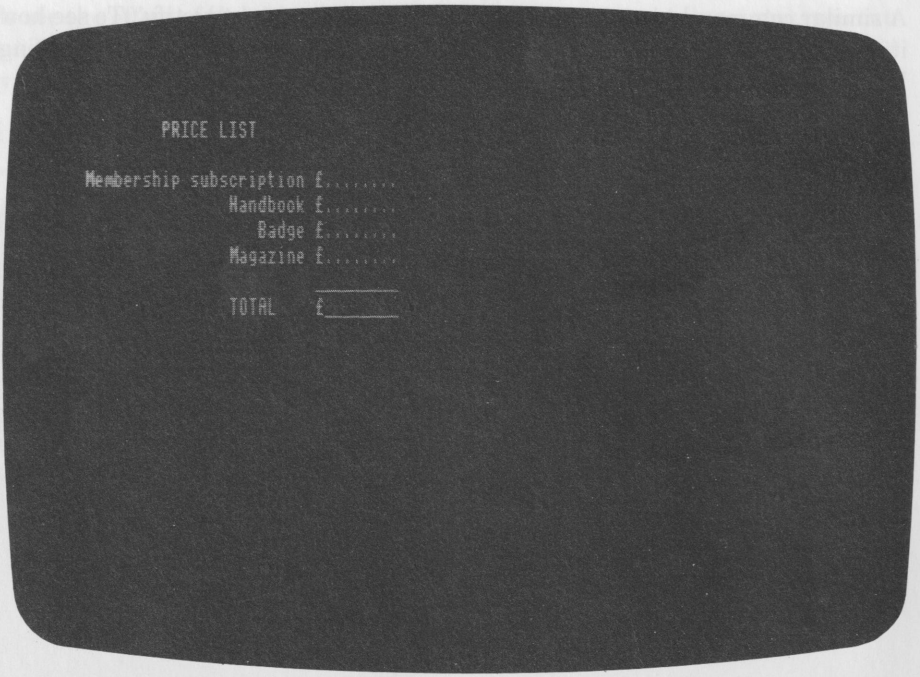
A similar command to CE is /RJ , which stands for 'right justify'. To see how it works type the following, entering the stored commands at the left using EDIT COMMAND as described above. Use a ruler of about 40 characters.



Switch to the command screen and type:

SCREEN **RETURN**

and you should see the heading centred (in relation to the ruler, of course, not necessarily the screen), and all the items below it justified right.



When you use the SCREEN command to look through a long piece of text, you will find that VIEW gives you a screenful at a time. To move on to the next screenful, press and release **SHIFT**. Notice that VIEW also shows you where the page breaks occur in your text. Page breaks are shown by gaps in the text.

Another stored command which you will need at an early stage in your word processing is PE – which stands for 'page end'. This is placed in the margin, using EDIT COMMAND in the same way as shown above. Its function is to tell the printer to move on to the next page at that point. VIEW will move the printer on to the next page in any case after it has reached an appropriate page length, but if you want to specify particular places for page breaks you should use PE. This is why it is useful to be able to see where the page breaks occur when you SCREEN the text. If you notice that a page break occurs in an awkward place such as in the middle of a table, you can alter this by inserting PE at the appropriate place.

We have just mentioned page length – how does the computer know what page length to use? In the absence of any instruction from you, VIEW assumes a page length of 66 lines. If this does not suit the length of the paper you are using, you can enter a stored command early on in your text like this:

Press EDIT COMMAND
The cursor moves into the left margin.
Type: `PL` **RETURN**
The cursor moves back into the text area and the characters `PL` remain in the left margin.
Type: `46`

The result should look like this:

```
PL 46
```

with `PL` in the stored command margin and `46` in the text area against it. This tells VIEW that you want to work to a page length of 46 lines. As soon as this length is reached, VIEW will issue an automatic 'page end' instruction to the printer, unless you forestall it with a `PE` instruction of your own.

There are many more stored commands and they are dealt with in detail later in this book. For the moment, try out the commands above for yourself, checking their effect with the `SCREEN` command.

Printing out from VIEW

As usual in this book, we have to assume that your equipment is connected correctly and working. You should ask your dealer to advise you on using printers with VIEW and make sure that the system is working to your satisfaction.

Once everything is set up correctly, printing from VIEW is very simple. You have only to issue the command `PRINT` **RETURN** and the contents of text memory are printed out. Similarly to print the contents of a file on disc or cassette you have only to type `PRINT` *`filename`* **RETURN**.

Before you get to that stage, however, you will have to make sure that the correct printer driver is loaded (see below).

Printer drivers

VIEW itself does not directly manage the printer. Instead it sends codes to a printer driver program which in turn sends other codes to the printer to produce the effect you have specified in your text and stored commands. This may seem an unnecessarily complicated way of doing things, but it has an important benefit: VIEW can print out on virtually any printer, with no modification. So if you have one kind of printer in the office and another at home, you should be able to use the facilities of each to the full. In fact VIEW itself contains a 'default' printer driver program, which automatically operates unless you load a special driver of your own. The default driver is adequate for straightforward printing on most printers.

But some printers are capable of more than ordinary, straightforward printing. Many offer underlined and bold type. Some offer superscripts, subscripts, and special characters. If you have a high quality daisy-wheel printer at least some of these effects will be available on it – but before you can use them with VIEW you need a printer driver program installed.

The *Printer Driver Generator* is a program designed to generate tailor-made printer drivers for VIEW. Before you buy a printer you should make sure that you can get a driver which controls its facilities to your satisfaction.

Loading your printer driver

The command to load a printer driver is:

`PRINTER` *`filename`* **RETURN**

The filename would normally be the name of the printer itself. For example the text of this book was composed and edited in VIEW and was first printed out as a draft on a Facit daisy-wheel printer. The printer driver was made by the *Printer Driver Generator*, and was given the name Facit. So the command to load this printer driver is:

`PRINTER Facit` **RETURN**

Setting up the printer

VIEW can print on any printer that will work with your computer. Before printing you should set up the printer in the usual way from VIEW's command-mode screen. Refer to the computer's User Guide for details of how to do this.

Printing out

Once everything is working correctly, you can print out in two ways:
- Printing out whatever text is currently held in the computer's memory.
- Printing from a file on the filing system.

To print from memory, switch to the command screen and type:

`Print` **RETURN**

To print you must name the file in the command:

`PRINT` *`filename`* **RETURN**

Printing out from a file does not affect the text held in memory. So you can take a break from editing the text in memory to print out some files, and then carry on editing the same text which will still be in memory.

Try printing out a few pieces of text to get used to the method.

Highlights

We mentioned earlier the possibility of using underlined type, bold type, and other special printer facilities with VIEW.

You have probably noticed by now the words HIGHLIGHT 1 and HIGHLIGHT 2 in the centre of the function key card. These refer to codes that may be inserted on either side of selected text, to indicate that this text is to be printed in a special way. Normally text marked with highlight 1 codes is printed underlined, and text marked with highlight 2 codes is printed bold.

To mark text with hightlight 1 codes, move the cursor to the first character and press HIGHLIGHT 1

Move the cursor to the character space after the last character in the text to be marked and press HIGHLIGHT 1 again.

In all modes except 7 the highlight 1 code is a minus sign on a white background. In mode 7 it is a minus sign.

Use the same procedure for highlight 2, except that your press HIGHLIGHT 2

In all modes except 7 the highlight 2 code is an asterisk on a white background. In mode 7 it is an asterisk.

Highlighted text looks like this in mode 3:

Text to be printed underlined is marked like this:

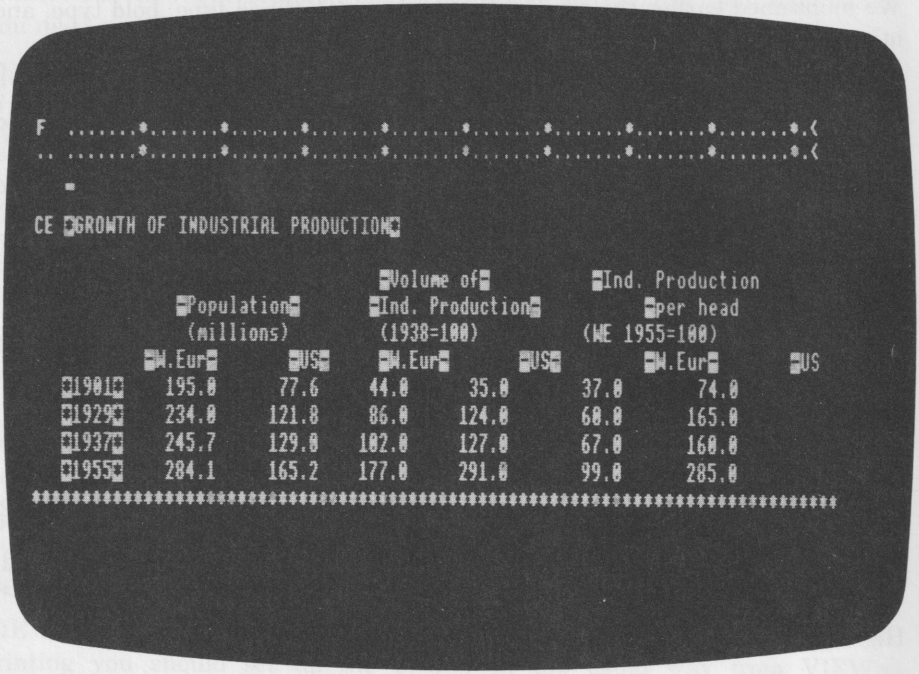
```
- underlined -
```

Text to be printed in bold type is marked like this:

```
* bold type *
```

You will naturally tend to use highlighted text for headings of chapters and tables. When you do this you will quickly become aware that the highlight characters occupy space on the text screen, but occupy no space at all when the text is printed.

This calls for some care. For example the following table looks very ragged on the text screen.



When printed out it looks like this. All the headings are in place and the columns aligned. This kind of thing takes some practice. The best general rule is to set out the table first, get it lined up as you want it and only then put in your highlight codes.

GROWTH OF INDUSTRIAL PRODUCTION

	Population (millions)		Volume of Ind. Production (1938=100)		Ind. Production per head (WE 1955=100)	
	W.Eur	US	W.Eur	US	W.Eur	US
1901	195.0	77.6	44.0	35.0	37.0	74.0
1929	234.0	121.8	86.0	124.0	60.0	165.0
1937	245.7	129.0	102.0	127.0	67.0	160.0
1955	284.1	165.2	177.0	291.0	99.0	285.0

There is, of course, much more to the highlighting of text than we have described in this chapter, and this will be dealt with later.

For the moment, feel free to experiment with all the printing facilities described so far. What about type that is both bold and underlined, for example? Only by trying things out and looking at the results will you become proficient.

Conclusion

This concludes the 'basic techniques' chapters of this book. By now you should have a good idea of what word processing in VIEW is all about – writing and editing text, moving and copying text, formatting, screen modes, saving, loading and printing.

In the chapters which follow, more rapid editing methods will be explained and further facilities described.

6 Moving the cursor

Using the arrow keys to move the cursor a character at a time horizontally or a line at a time vertically is precise but slow. More rapid movement is available using **SHIFT** with the arrow keys, and more rapid movement still using **CTRL**. In addition there are function key commands for cursor movement.

Horizontal cursor movement

– Arrow keys alone: the cursor moves one character space at a time right or left, with continuous movement if the key is held down.

– **SHIFT** arrow keys: the cursor jumps along the line right or left, resting on the first character of each word, and the position after the last character of the line, before moving on to the following line.

– **CTRL** arrow keys: the cursor jumps to the first character, or the position after the last character of the line.

– BEGINNING OF LINE: the cursor jumps to the first character of the line.

– END OF LINE: the cursor jumps to the position after the last character of the line.

(The two function key commands and the arrow key commands with **CTRL** duplicate each other. Use which you like.)

Vertical cursor movement

– **RETURN**: the cursor moves to the beginning of the next line down in the main text area, moving down the row of asterisks at the bottom of your text if necessary.

– Arrow keys alone: the cursor moves one line at a time up or down, with continuous movement if the key is held down.

– **SHIFT** arrow keys: the cursor jumps the height of the screen, up or down.

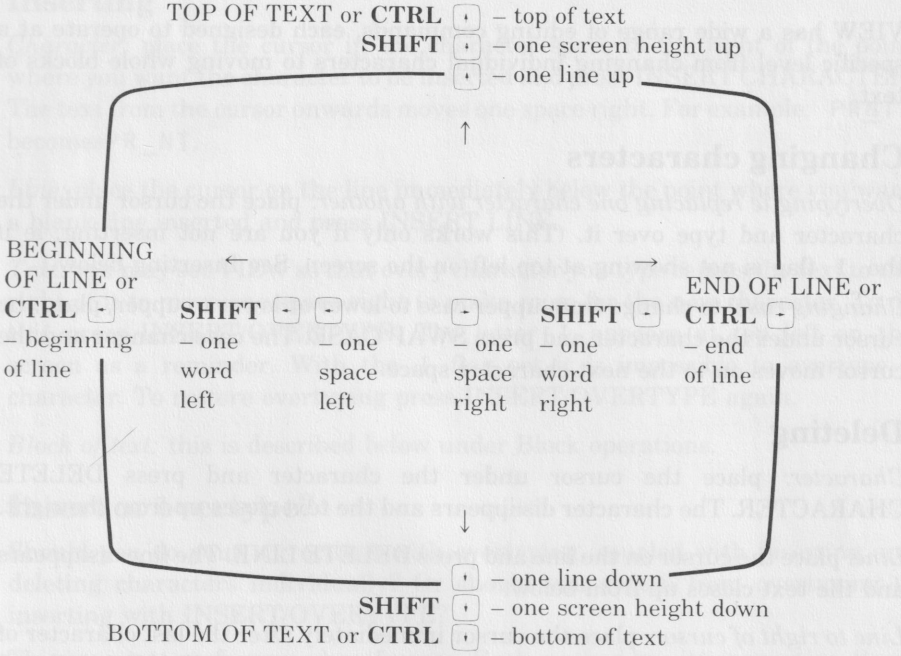
– **CTRL** arrow keys: the cursor jumps to the top or the bottom of the text.

– TOP OF TEXT: the cursor jumps to the top of the text.

– BOTTOM OF TEXT: the cursor jumps to the space following the last character of the text.

(The two function key commands and the arrow key commands with **CTRL** duplicate each other. Use which you like.)

The following diagram summarises the commands.



7 Editing text

VIEW has a wide range of editing commands, each designed to operate at a specific level from changing individual characters to moving whole blocks of text.

Changing characters

Overtyping ie replacing one character with another: place the cursor under the character and type over it. (This works only if you are not inserting, ie if the `I` flag is not showing at top left on the screen. See Inserting below.)

Changing case: to change from upper case to lower or lower to upper, place the cursor under the character and press SWAP CASE. The case changes and the cursor moves on to the next character space.

Deleting

Character: place the cursor under the character and press DELETE CHARACTER. The character disappears and the text closes up from the right.

Line: place the cursor on the line and press DELETE LINE. The line disappears and the text closes up from below.

Line to right of cursor: place the cursor immediately after the last character of the line that you wish to preserve and press DELETE END OF LINE. All characters to the right of the cursor disappear. For example, all words after `space` will be deleted in the line below.

```
In ViewStore, the space_that the
```

Line up to named character: place the cursor immediately after the last character to be preserved and press DELETE UP TO CHARACTER. The characters `CH` appear at top left on the screen. Then press the character up to which you wish to delete the line. After the characters are deleted the line closes up from the right. For example, in the following line, an asterisk has been inserted to mark the position up to which the line is to be deleted. If you now press DELETE UP TO CHARACTER followed by `*`

```
In ViewStore, the_space that*the
```

the words `space that` are deleted and the line closed up like this:

```
In ViewStore, the_the
```

Note that the character specified when using DELETE UP TO CHARACTER can be a space, a tab character or a highlight character.

Block of text: this is described below under Block operations.

Inserting

Character: place the cursor in the character space to the right of the point where you want the character to be inserted and press INSERT CHARACTER. The text from the cursor onwards moves one space right. For example: `PRNT` becomes`PR_NT`.

Line: place the cursor on the line immediately below the point where you want a blank line inserted and press INSERT LINE.

Text: you may set VIEW so that every character you type causes the text to the right of it to move one space right to make room for the new character. To do this press INSERT/OVERTYPE. The letter `I` appears at top left on the screen as a reminder. With the `I` flag set it is impossible to overtype a character. To restore overtyping press INSERT/OVERTYPE again.

Block of text: this is described below under Block operations.

Insert or overtype?

Should you do your corrections with overtyping, coupled with inserting and deleting characters individually? Or should you switch from overtyping to inserting with INSERT/OVERTYPE?

This is a matter of personal preference. Each method has its supporters. Some people abandon overtyping altogether and keep the `I` flag set all the time. Some use either method as the mood takes them.

While you are learning to use VIEW you should make yourself thoroughly familiar with all its facilities, even if you later select those which suit your work best.

Line operations

Splitting a line: place the cursor under the character which you intend to be the first character of the following line and press SPLIT LINE. For example, the line below

```
The Picture Maker package
```

would be split at the first character of `package`, like this:

```
The Picture Maker
package
```

Joining lines: the two lines must be adjacent. Place the cursor on the upper line and press JOIN LINES.

In practice you may ofter prefer to join lines and format them at the same time by pressing FORMAT PARAGRAPH – but you should remember that this will affect subsequent adjacent lines too.

Block operations

In order to move, copy or delete a block of text, you must first set up markers to tell VIEW which block of text you mean.

To mark a block of text, place the cursor at the first character of the block to be marked.
Press SET MARKER.
The characters `MK` appear at top left on the screen.
Press: `1`
A white block appears at the first character.

Move the cursor to the space after the last character of the block to be marked.
Press SET MARKER.

The characters `MK` appear at top left on the screen.
Press: `2`
A white block appears above the cursor.

Moving a block of text: place markers as described above. Move the cursor to the point where you want the block to be moved to. Press MOVE BLOCK. The block is moved, the rest of the text is rearranged accordingly, and the markers disappear.

Copying a block of text: place markers as described above. Move the cursor to the point where you want the copy. Press **COPY**. The block is copied, the rest of the text is rearranged if necessary, but the markers remain in place so that you can make many copies of the same block if you wish.

Deleting a block of text: place markers as described above. Press DELETE BLOCK. The block is deleted, the text is closed up, and the markers disappear.

Using markers

As described above, markers 1 and 2 are shown in the text as a white block. Markers 3, 4, 5 and 6 also exist, but these are invisible. You can use them with GO TO MARKER.

To set a marker:
First place the cursor where you want the marker.
Press SET MARKER
The characters `MK` appear at top left on the screen.
Press *number* (Numbers may be from 1 to 6.)
The message `Marker(s) set` *number* appears on the command screen.

To go to markers:
Press GO TO MARKER
The characters `MK` appear at top left on the screen.
Press *number*

To clear all markers:
Switch to the command screen and type:

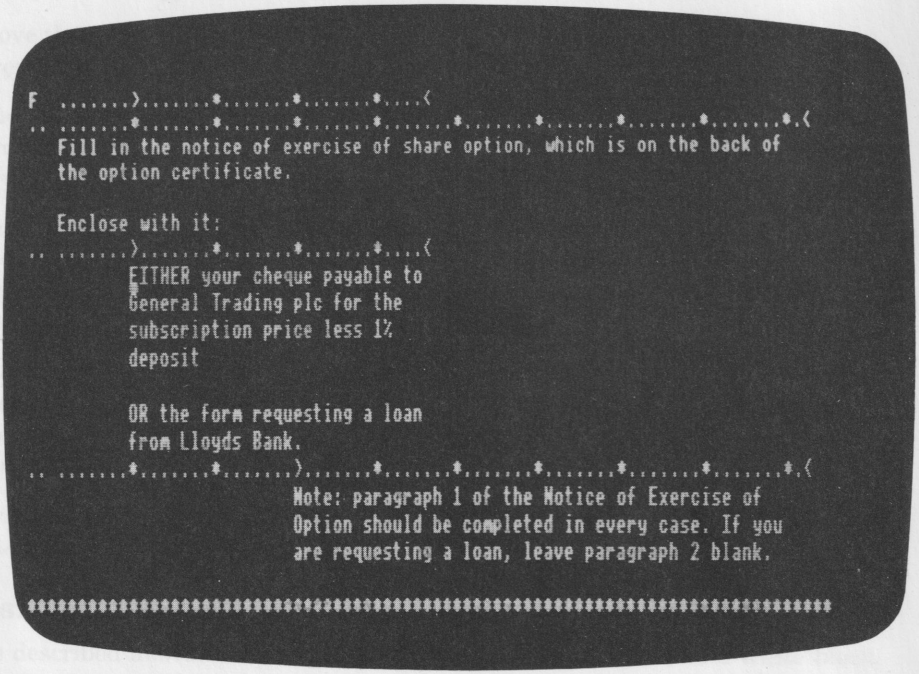
`CLEAR` **RETURN**

8 The ruler, tabbing and formatting

The ruler controls the width and tabbing of the column of text typed under it.

When a line of text is typed, the ruler above it regulates the length of the line. When formatting is active (F is shown at top left on the screen), words which overshoot the right margin are transferred whole to the next line.

In the following screen, three line lengths are used. Notice how the ruler at the top of the screen matches the current ruler ie the ruler above the cursor.



Notice also the left margin signs (>). All the rulers shown here have been specially created by the user to place the text where he wanted it.

The ruler at the top of the screen

The ruler that appears on the top line of the text screen is a copy of the current ruler. This top ruler is always present on the text screen and serves to remind you which ruler is active where the cursor is. *Remember that VIEW always obeys the ruler above the line that the cursor is on.*

In two respects, the ruler at the top of the screen is unlike the others:

1. You cannot move the cursor on to the ruler at the top of the screen and so cannot directly alter it.

2. Instead of two dots in the left margin it has 'flags', ie the following characters may be present:

`F` – Formatting is switched on: words which overshoot the end of the line are transferred to the following line.

`J` – Justifying is switched on: spaces in lines are adjusted so that the lines are all the same length.

`I` – Inserting is switched on: text already on the screen opens up to receive new text.

`M` – Margins are not active. This flag appears, for example, if you move the cursor into the area to the left of a left margin stop (this is described later) or on to a user-defined ruler.

The make-up of rulers

```
FJ .......*.......*.......*.......*.......*.....<

.. .....>.....*.....*.....*.....*.....B...*.<
```

The asterisks are tab stops and can be used in much the same way as tab stops on a typewriter: press **TAB** and the cursor moves forward to align itself with the next asterisk.

The two dots in the left margin are a sign that the line is a ruler.

`>` and `<` are margin stops.

If a left margin stop is set, when you press **RETURN** the cursor returns only as far as that stop. (You can in fact move it further left, and we shall deal with this later.)

The `B` or `b` character can be placed in a ruler to cause a bleep when the cursor passes it. This is rather like the bell on a typewriter, except that it has nothing to do with margins.

Creating rulers

You may create up to 128 rulers in any document. It is in fact good practice to create a ruler when you start a new document, so that if the standard ruler changes, your text is not reformatted. There are three ways of creating your own rulers:

– *Copy the current ruler:* to copy the ruler that is active where the cursor is, press **SHIFT COPY**. Copying the current ruler is often a convenient way of getting a ruler which you can modify.

– *Put a copy of the default ruler on to the screen:* press RULER. This gives you a copy of the default ruler for the screen mode you are in. You can then edit this ruler.

– *Make up your own ruler:* enter the tabs, dots and margin stop(s) and press MARK AS RULER to convert the line into a ruler.

Some caution is advisable when creating or modifying rulers. In theory the minimum requirements for a ruler are the two dots in the stored command margin. So the following would be perfectly normal rulers.

```
..  This is a ruler This is a ruler This is a ruler <

..                                                   <
```

In practice this kind of thing is asking for trouble – such rulers are very easy to delete by mistake or to edit in ways that will make your text look very peculiar. So make your rulers look like rulers.

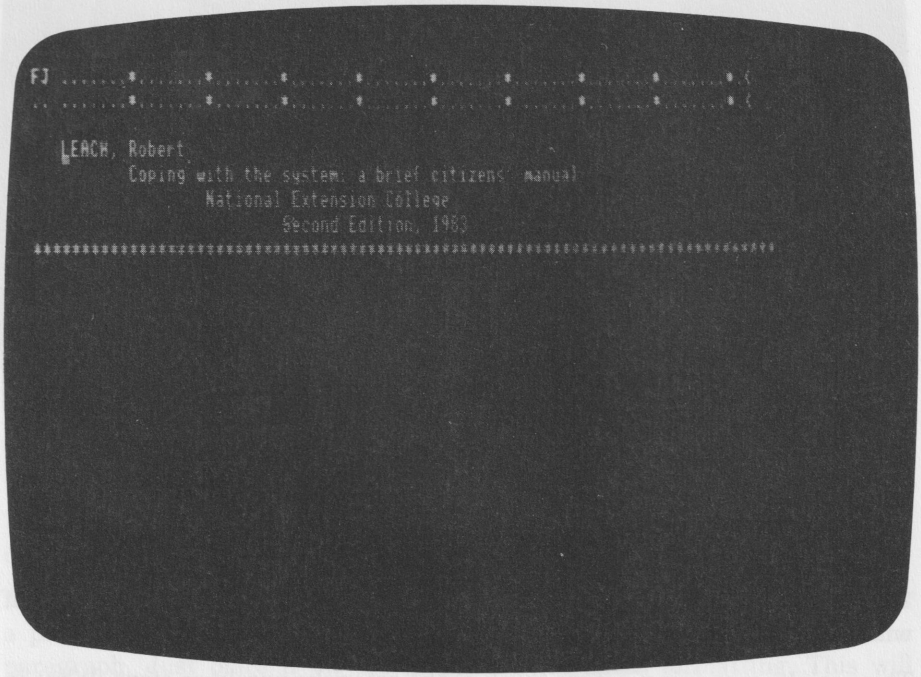
Deleting rulers

There are two ways of deleting a ruler. One is to place the cursor on the ruler and press DELETE LINE.

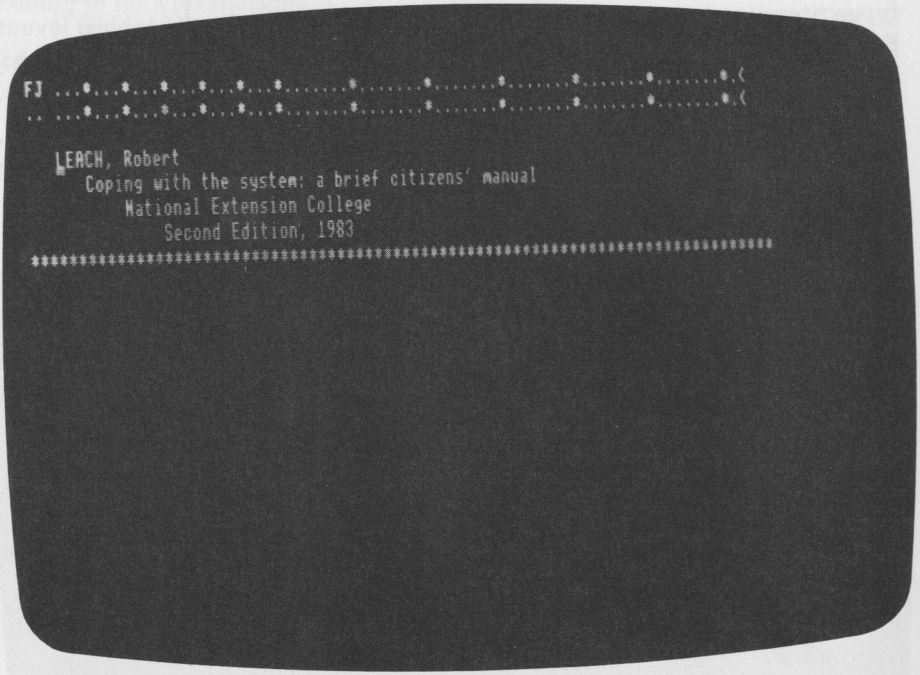
If you do not want to delete the line and cause the text to close up, you can place the cursor at the left end of the ruler and press DELETE END OF LINE. This leaves the two dots in the command margin. You should delete these with DELETE COMMAND.

Tabbing

Although the tab stops in VIEW rulers have much in common with those on a typewriter, word processed text is not by any means fixed in its tabbed layout. For example the following screen shows an entry in a booklist in which no tab is used in the first line, one in the second, two in the third and three in the last.



You can, however, edit the ruler to change the tab spacings like this:

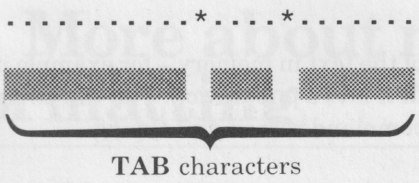


When you move the cursor down off the ruler, the text instantly aligns itself with the new tabs.

This may be surprising – if we think of the asterisks purely as stops. What in fact is happening is that VIEW is inserting tab characters in the text. These, like spaces, are invisible characters, but you can detect their presence quite easily by deleting them.

Try this.

Put a ruler on the screen by pressing RULER. Press **RETURN** to get the cursor to the beginning of a line. Press **TAB** three times and type in a word. Move the cursor back to the beginning of the line using the arrow key, and press DELETE CHARACTER. The text moves back to the left margin in three jumps, as the three tab characters are deleted.



The size of the tab characters is set by the tab stops on the ruler, so when you change the setting of the asterisks on the ruler, the text under it is re-arranged too. This can be very useful when designing tables. First set the tabs on the ruler approximately. Then type in the table. Finally adjust the tab stops on the ruler until the table looks right. Take care to make sure that all the tabs have been inserted into the table.

Formatting

Formatting limits the length of lines to the current ruler.

Its effect is shown when the word you are typing at the end of the line overshoots the margin and is transferred whole to the next line.

Formatting is normally on, and this is confirmed by the character F at the left of the top ruler. To turn formatting off press FORMATTING. (The same key turns it on again.)

To format a paragraph after it has been typed, place the cursor on the top line of the block and press FORMAT PARAGRAPH.

It is perhaps worth mentioning that if you only need to format the lower half of a paragraph there is no point in placing the cursor on the top line of the paragraph. Just place it on the first line that needs formatting. This will probably be the first line on which your corrections began.

When VIEW starts to format a paragraph it goes on to the end of that paragraph. To VIEW, a paragraph means something quite precise: it is a block of adjacent lines ending in one of the following:

- A blank line
- A line with a tab on it.
- A line beginning with a space.
- A line with a stored command on it.
- A line with text in the left margin (see chapter 9).
- A ruler.
- The end of the document.

So you should beware of formatting something which you consider to be a paragraph, but which does not meet VIEW's definition of a paragraph. If you do you may find that VIEW takes the formatting further than you intend.

Global formatting

You may wish to re-format the whole of the text in memory – for example if you decide to change the column width to fit it into a publication. You can do this by changing the ruler, switching to the command screen and typing:

`FORMAT` **RETURN**

It is worth looking through your text before you do this, to see if there is any text you do not wish to format, such as text with headings in the right margin.

If you wish to format a substantial part, but not all of your text with the global format command, you can do so by setting markers 1 and 2 before and after the part to be formatted, switching to the command screen, and typing:

`FORMAT 1 2` **RETURN**

The method of setting markers 1 and 2 is described in chapter 7.

Justification

When a word overshoots the right margin and is transferred whole to the next line, as mentioned above, one of two things may happen.

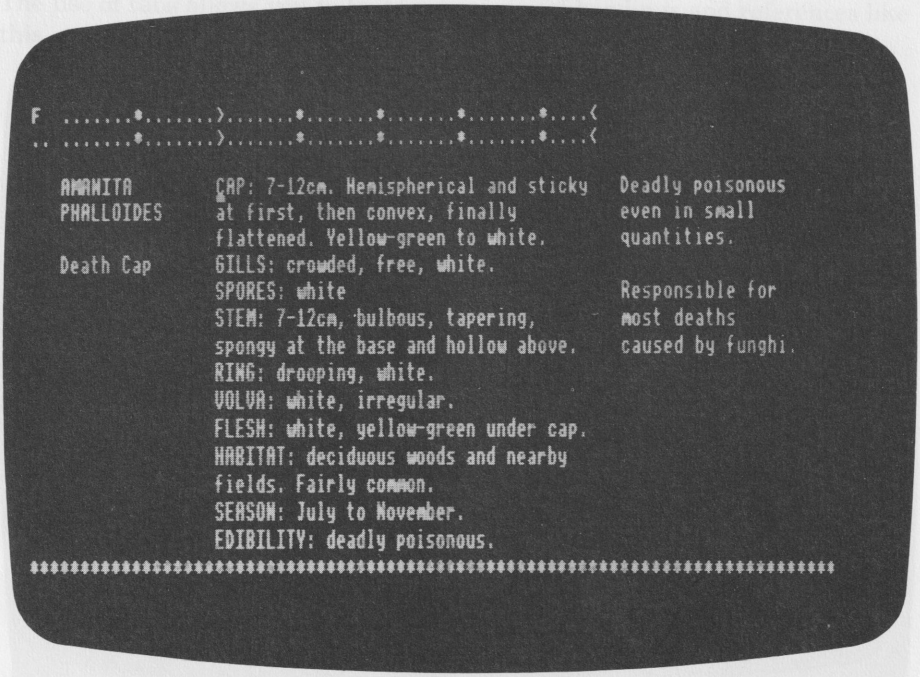
If justification is switched on, VIEW adds spaces to the spaces between the words such that all lines are of the same length.

If justification is not switched on, the spaces are not adjusted and the lines will be of uneven length.

Justification is on when you enter VIEW. To turn it off, press JUSTIFICATION. (The same key turns it on again.)

9 More about margins and formatting

Most of the text you type into VIEW will be regulated by rulers in the ways described in the last chapter. It is also useful, however, to be able to place text outside the ruler – side headings and notes in a report, for example, or authors' names in a publisher's catalogue. The following screen shows another such example.



Text beyond the left margin stop

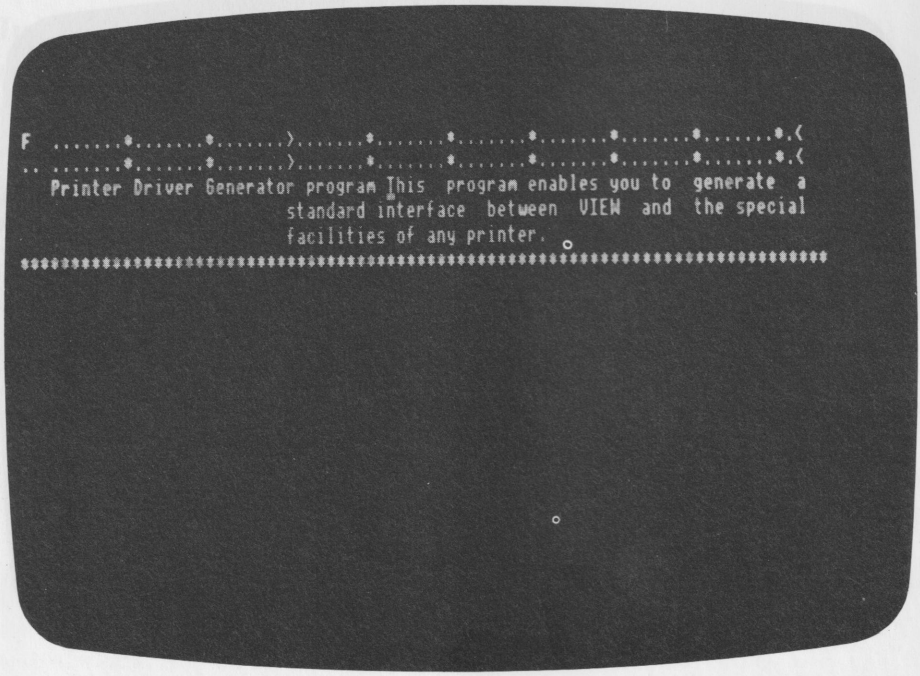
To set text with side headings and notes, first create a ruler with a left margin stop set so as to allow room for text on the left. The method of creating such a ruler is described in the previous chapter.

Type in the text and make sure it is formatted to your satisfaction. Place the cursor on the first character space of the line on which you want text in the left margin. Press the left arrow key to move the cursor into the left margin.

Then enter your text on that line in the usual way, using the normal editing commands to modify it. Only when you approach to within a character space of the left margin stop will your text affect the text between the margins.

The left margin heading is not affected by formatting the main text.

Note that VIEW allows you not only to set text in the left margin as in the screen example above, but to make headings that start in the left margin and go on continuously into the text area, overhanging the main column of text. In such a case, formatting will affect only the main text; the heading will be left intact.



Text beyond the right margin stop

In order to set text beyond the right margin stop, should you wish to do so, you will have to turn VIEW's formatting off.

Press FORMATTING

Use the arrow keys to position the cursor, and type in your text.

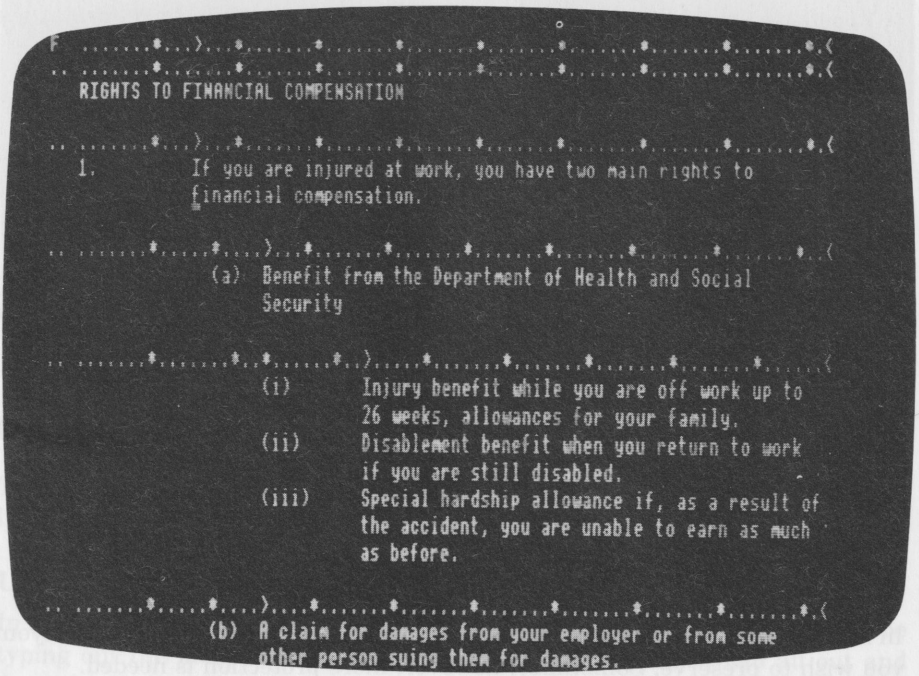
When you have finished, press FORMATTING to turn formatting on again.

You should remember, however, that any attempt to re-format the text after this will result in whatever you have entered beyond the right margin stop being caught up with the rest of the text a line at a time – ruining all your carefully constructed layout.

Working beyond the left margin stop

It is possible to operate beyond the left margin stop with much greater freedom than might seem likely. Most commands, including **TAB**, INSERT CHARACTER, DELETE CHARACTER and HIGHLIGHTs function normally in this area.

The use of tabs allows you to build up a series of headings and references like this.



Note how the margin stops and tabs are aligned so as to give a consistent layout.

In text such as this, the area to the left of the left margin stop is in fact a left margin tab character – which you can prove by deleting it and so causing the text to move left. When you type the first character on a line, VIEW checks for a left margin stop. If you have not set one, VIEW places the character in the first character position on the line. If you have, VIEW inserts a left margin tab character filling the space between the leftmost position on the line and the left margin stop.

VIEW leaves a gap between the end of the heading and the beginning of the main text.

If you type a piece of text with a left margin stop set, and then remove the stop from the ruler, the text will align to the left, but with one space between the text and the leftmost position on the line. This is because the left margin tab is still there, reduced to one character space now that the margin stop has been removed.

When you re-format the text, the space to the left disappears. Similarly if you define a left margin for a piece of text which had none before, you will need to re-format the text. In fact you should make it a rule always to re-format the text after adjusting the margin stops.

Protection from formatting

As described earlier, text may be formatted by placing the cursor on the first line that need formatting and pressing FORMAT PARAGRAPH. It may also be formatted by switching to the command screen and typing:

```
FORMAT    RETURN
```

As mentioned in the last chapter, once VIEW starts to format a paragraph it goes on until it comes to any one of the following:

- A blank line.
- A line with a tab on it.
- A line begining with a space.
- A line with a stored command on it.
- A line with text in the left margin.
- A ruler.
- The end of the text.

In most cases these should be sufficient to prevent the formatting of any layout you wish to preserve. Sometimes, however, more protection is needed.

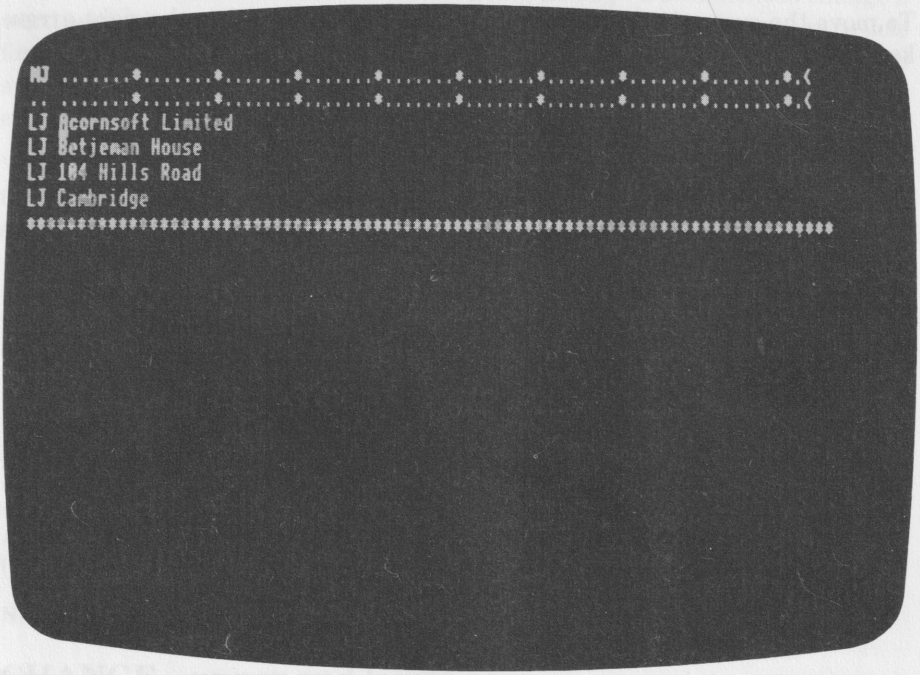
Using a stored command

Addresses, for example, often begin at the leftmost position on the screen and would not preserve their layout if the text were formatted unless specially

protected. The simplest way is to use one of VIEW's stored commands: `LJ` – meaning 'left justify'.

Place the cursor on the line to be protected.
Press EDIT COMMAND
The cursor moves into the stored command margin.
Type: `LJ` **RETURN**

An address protected in this way looks like this:



The `F` flag has changed to `M` because the cursor is on a stored command line.

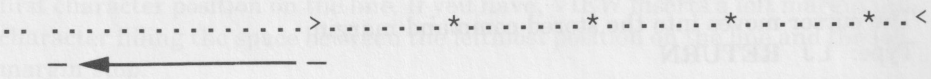
Using a ruler

In some cases you may find this method inconvenient – for example you may be typing out the labels for an illustration, intending to print them all out and draw the illustration on to the print-out. If some of the labels start on the left of the line, the layout could be disturbed by formatting.

In this case a useful device to prevent formatting is to create a ruler above your layout with its right margin stop removed.

Margins

MARGINS has a similar effect to the left and right arrow keys when you have set the left margin stop so as to provide a left margin.



To move the cursor to the first character space in the left margin, press either the left arrow key or MARGINS.

To move the cursor back into the main text area press either the right arrow key or MARGINS.

10 CHANGE . . . REPLACE . . . SEARCH

CHANGE

From time to time we all face the task of going through a document to change a word every time it occurs. For example, you may be editing an American article for English publication, and you want to change everything over to English spelling. The chances of missing one use of 'color' or 'center' are quite high, especially if the article is long.

VIEW takes care of such problems with the `CHANGE` command. To change the spelling from 'color' to 'colour' throughout the text, switch to the command screen and type

`CHANGE color colour` **RETURN**

Having completed the task, VIEW gives you a message such as:

```
17 string(s) changed
```

In computer terms a string is a series of characters such as occur in words and phrases. So this message means that VIEW has made 17 changes throughout the text, in accordance with your instructions.

Alternatively if no instance of the word you have specified occurs in the text (perhaps because you have spelt the word wrongly) you get the message:

```
No string found
```

CHANGE – upper and lower case

If you try out a few `CHANGE` operations you will quickly discover that it makes no difference whether you type

`CHANGE center centre` **RETURN**

or `CHANGE Center Centre` **RETURN**

or `CHANGE CENTER centre` **RETURN**

VIEW ignores the case of the characters you type in the command. Instead it very usefully imitates the case of the characters it finds in the text. Thus if the original is `center` the replacement becomes `centre`; if `Center` it becomes `Centre`; and if `CENTER` it becomes `CENTRE`.

Usually this is exactly what you want. There are times, however, when you need to specify the case of some of the characters in the substituted word. Sometimes the only reason you want the change is for a change of case. For example, you may be writing about an antibiotic called 'erythromycin'. You have nearly finished your article when you find that this is a proprietary name, and should be written 'Erythromycin'.

To make a correction like this you need to turn off VIEW's facility to copy the case of the original. This facility is known as 'folding', and you can turn it off by switching to the command screen and typing

`FOLD OFF` **RETURN**

After this you can issue the instruction:

`CHANGE erythromycin Erythromycin` **RETURN**

and the change will be made. To turn folding on again type

`FOLD ON` **RETURN**

If you forget whether folding is off or on, type

`FOLD` **RETURN**

and VIEW will tell you.

CHANGE – words and phrases

The space between the words of a `CHANGE` command has a definite function: it signals to VIEW that you want the word before the space changed into the word after it. To replace one group of words with another, therefore, requires a slightly different method.

The simplest way is to use the slash sign (`/`) instead of a space. For example, suppose you are editing an article on ornithology and the author is writing about 'dunnocks'. You happen to know that 'dunnock' is another name for the 'hedge sparrow' and you feel that the latter is the more familiar word. So you make the change like this:

`CHANGE/dunnock/hedge sparrow/` **RETURN**

The space or slash in the `CHANGE` instruction is known as a 'delimiter', since its function is to show where one thing ends and another begins. The first character after `CHANGE` indicates to VIEW which delimiter is being used.

You may use any of the following characters as a delimiter, provided you use it consistently through the instruction:

```
! " # $ % & ' * + , = . /
```

CHANGE – parts of words

The `CHANGE` command interprets your instructions quite precisely. Suppose you are writing a company report and your company has recently been reorganised: instead of 'departments' you now have 'divisions'. No problem:

```
CHANGE department division
```  **RETURN**

Not only will `department` be changed to `division`, but `departments` will be changed to `divisions` and `departmental` to `divisional`. In this case no doubt that is the result you want, but you should use `CHANGE` with care. To prevent any change but the one you have specified, you can give the command in the form:

```
CHANGE/ department / division /
```  **RETURN**

That is: change department with a space before and after to division with a space before and after. Note that if you think either of these words is likely to appear with a punctuation mark after it, you will have to cater for this.

CHANGE – part of text only

Sometimes you wish to change a word or phrase in part of a document only. This can be done by setting marker 1 before the area in which you want the change, and marker 2 after it. (For setting markers, see chapter 7.)

So, to continue our ornithological example, if you decide that in a more technical section of the article, the name 'hedge sparrow' should be replaced by the scientific name, you have only to set markers 1 and 2 before and after that section and type

```
CHANGE/hedge sparrow/prunella modularis/12
```  **RETURN**

CHANGE – special characters

The `CHANGE` command can specify a very wide range of characters. Apart from all the characters normally used in text, you can `CHANGE` the following. The 'hat' or circumflex symbol must be used in these cases.

`^S` – hard space (a space character which you have inserted)
`^Z` – soft space (a space character which VIEW has inserted, when justifying text)
`^T` – tab
`^C` – Carriage Return character

^L – left margin tab
^- – highlight 1
^* – highlight 2
^? – wild character
^^ – ^character

It may appear at first glance that you are not likely to have much use for some of these. However, suppose you are editing a list of companies. Your house style dictates that when companies are referred to in text the form 'Limited' is used, but where used in tables the word appears abbreviated as 'Ltd'.

The tables have already been typed, with `Limited` typed in full. Each company name is followed by statistics in the form:

```
Products Limited        £12,345        £123,000        13.22
```

You know that in this tabular layout the word `Limited` is always followed by a tab character, so you can specify in your change command that only in those cases where `Limited` is followed by a tab should it be changed to `Ltd`, like this:

`CHANGE Limited^T Ltd^T` **RETURN**

The **RETURN** character may also be used in the `CHANGE` string. Suppose you are designing a form which asks for details of all the members of an organisation in the form:

```
Membership no.
Name
Address
```

and you want to convert this into

```
Name
Address
Membership no
```

Since, although you cannot see it, there is a Carriage Return at the end of each of these lines. So you can use the instruction:

```
CHANGE/Membership no.^CName^CAddress^C/Name^CAddress^CMembership no.^C
```

followed by **RETURN**

The wild character (`^?`) is quite straightforward to use. It stands for any character or characters. Suppose you are editing a text in which you find the word `definitely` spelt `definately`. When you issue the `CHANGE`

command it occurs to you that the typist may have made other mistakes with the same word. So you specify a wild character like this:

```
CHANGE defin^?tely definitely  RETURN
```

and spellings such as `definetely` and `definately` would all be corrected.

If you wanted to delete all the highlight 2 codes from your document, you would type:

```
CHANGE/^*//
```

ie replace all highlight 2 codes with nothing, as signalled by two delimiters with nothing between them.

REPLACE

The `REPLACE` command is used when you want to change some of the words only, and want to make a decision on each one. To return to our ornithological example, suppose you wanted to change `peewit` to `lapwing` – another synonym.

In most cases you would be safe to use the `CHANGE` command, but you know that somewhere in the text you have written: `The peewit is also known as the lapwing ...`, and here you will need to make more extensive changes to the sentence.

In this case you should use:

```
REPLACE peewit lapwing  RETURN
```

VIEW switches to the text screen and the first occurrence of `peewit` is signalled by the cursor being placed on the first letter and by the letters `RP` being shown in reverse video at the top left corner of the text screen.

If you do not want that word replaced, press `N`. If you want it replaced, press `Y`. If you want it replaced, but with folding disabled, press `O`. VIEW then moves on to the next occurrence of the word, until you have made your decisions on them all.

The `O` option above meaning 'yes, but with folding turned off' could be used, for example, if you were editing an article in which the writer had been careless about the use of capitals – perhaps using `nation` and `Nation` at random throughout the text. You decide in favour of lower case.

But you also decide to change the word `nation` into `government`. In most instances, therefore, you will be changing `nation` or `Nation` into

government, but in some instances (eg first words of sentences) you will want to use Government. The only way to do all this is by using

REPLACE nation government **RETURN**

So when VIEW asks you about nation or Nation you will normally press O and change them all to government; but if it lights on a sentence beginning

National decisions concerning foreign policy ...

you will press Y so that VIEW imitates the case and changes it to Governmental decisions ....

You may use the following facilities with REPLACE in the same way as described above under CHANGE:

- FOLD.
- Setting markers to limit the effect of REPLACE to part of the text.
- Special characters, including the wild character.

SEARCH

If you want to find a particular word but do not want to change it, or cannot predict how you will change it until you see it, the most useful command is SEARCH. This is given in the command screen in the same way as CHANGE:

SEARCH divisional **RETURN**

SEARCH/prunella modularis/ **RETURN**

VIEW changes to the text screen and the cursor rests on the first occurrence of the word named. To find the next, press **CTRL f1**-NEXT MATCH.

SEARCH can be used with FOLD in the same way as CHANGE, to locate only those instances of the word or phrase with a particular arrangement of upper and lower case characters.

SEARCH – the 'wild search' facility

This works in the same way as described under CHANGE. Suppose you want to find 'relevant' but you cannot remember whether it is spelt 'relevant' or 'relevent'. You can find it with:

SEARCH relev^?nt **RETURN**

SEARCH – other facilities

You may use the special characters described under CHANGE in specifying the target for your SEARCH.

You may limit the area of search with markers as described under CHANGE.

You may turn off folding and so include a particular arrangement of upper and lower case characters in your SEARCH.

Acornsoft's ViewSpell can also be used to check the spelling of text which has been composed on the VIEW word processor.

In principle, the method is to load a VIEW file into ViewSpell and tell ViewSpell to check it. The words in your file are then compared with the words in dictionaries and ViewSpell makes a list of any words in your text that are not in the dictionaries.

You can then display this list of unfound words on the screen or print it out. You can add any of the words displayed to your own user dictionary for future use. You can display the original VIEW file to remind yourself of how you used the words; and you can tell ViewSpell to produce a special version of the VIEW file with marks beside all the unfound words.

If you use specialised vocabulary, some of the words you use may not be in the ViewSpell master dictionary. You can get these checked by creating a user dictionary to supplement the master dictionary and you can add to this at any time.

You can also use the master dictionary to check spelling directly, by telling ViewSpell to search for a word. It you are not sure of the spelling, you can still search, using only the letters you know – a facility of interest to crossword puzzlers as well as writers.

11 Page layout and stored commands

VIEW's stored commands are all entered in the stored command margin on the text screen, using EDIT COMMAND. We have already used some of them: PE-PAGE END, CE-CENTRE TEXT, RJ-RIGHT JUSTIFY.

Stored commands have no effect on the text at all until it is printed, although you can often check their effect by simulating a print-out on the screen with the SCREEN command.

Entering a stored command

To enter a stored command such as PE, move the cursor to the line where you want the command.
Press: EDIT COMMAND
The cursor moves left into the stored command margin.
Type: PE **RETURN**

If you type a stored command wrongly, just type it again. Until you press **RETURN** and the cursor moves into the text area, the characters do not become a stored command.

Deleting a stored command

Place the cursor on the line with the stored command on it.
Press DELETE COMMAND

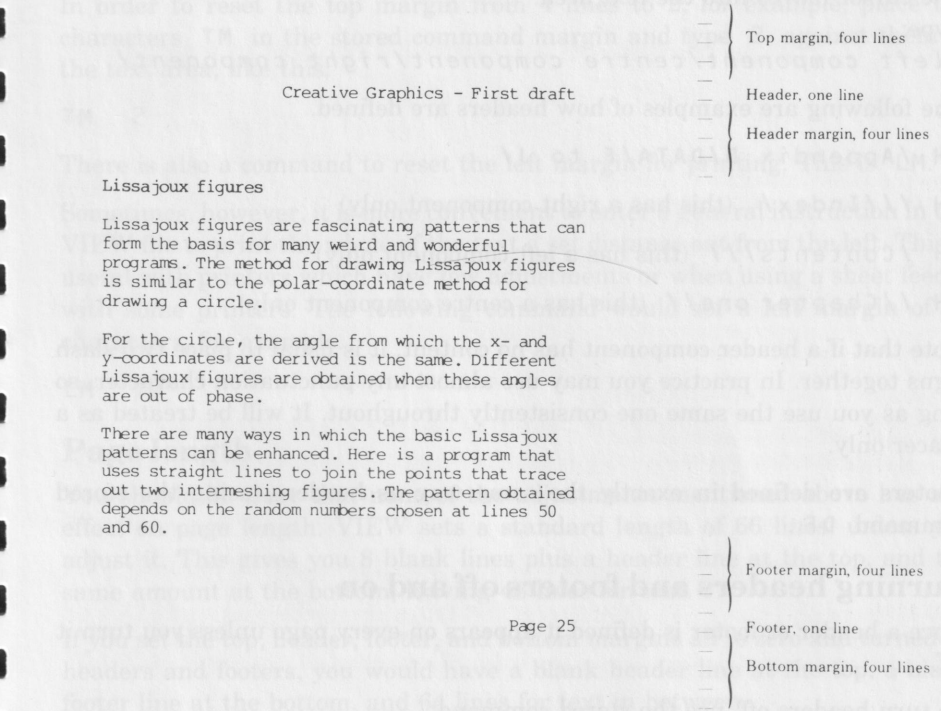
The VIEW page

Before dealing with stored commands in detail it is necessary to appreciate the way in which VIEW organises the printed page.

On the text screen, text is in a continuous, unpaged column – which is very convenient for the kind of editing we have been describing so far. When printing out, VIEW organises the column of text into pages, assuming a page length of 66 lines.

So every 66 lines, unless you give different instructions, VIEW moves to the start of a new page.

You can, of course, modify this by inserting a PE-PAGE END instruction of your own before the 66 lines are up. Within that page length, however, VIEW makes further subdivisions, like this:



So within the total page length of 66 lines VIEW allows:

- A top margin of 4 lines.
- A header line (whether or not a header actually occupies it).
- A header margin of 4 lines.
- A text area.
- A footer margin of 4 lines.
- A footer line (whether or not a footer actually occupies it).
- A bottom margin of 4 lines.

Headers and footers

You may define a header and a footer, so that every page of your document has the same header and footer printed on it automatically. Headers and footers have left, centre, and right components. The maximum length of a header or a footer is 63 characters.

To define a header, place the cursor before the page on which you need the header and press EDIT COMMAND
The cursor moves into the stored command margin.
Type: `DH` **RETURN**
The cursor moves into the text area.
Type:
`/left component/centre component/right component/`

The following are examples of how headers are defined.

`DH /Appendix D/DATA/F to J/`

`DH ///Index/` (this has a right component only)

`DH /Contents///` (this has a left component only)

`DH //Chapter one//` (this has a centre component only)

Note that if a header component has no content, it is usual to place two slash signs together. In practice you may use almost any punctuation character, so long as you use the same one consistently throughout. It will be treated as a spacer only.

Footers are defined in exactly the same way as headers, using the stored command `DF`.

Turning headers and footers off and on

Once a header or footer is defined it appears on every page unless you turn it off.

To turn headers off, use the stored command:

`HE OFF` (that is `HE` in the command margin, and `OFF` in the text area beside it)

To turn headers on again, use:

`HE ON`

To turn footers on and off in the same way use the command: `FO`

When headers or footers are turned off, the printer leaves a blank line where the header or footer would have been.

Setting page margins for printing

The top margin, header margin, footer margin and bottom margin are initially set by VIEW at 4 lines each, but all can be adjusted with stored commands. The commands to adjust each of them are as follows.

TM – top margin
HM – header margin
FM – footer margin
BM – bottom margin

In order to reset the top margin from 4 lines to 2, for example, place the characters TM in the stored command margin and type 2 against them in the text area, like this:

```
TM   2
```

There is also a command to reset the left margin for printing. This is LM.

Sometimes, however, it is more convenient to enter a general instruction in the VIEW file to print the whole of the text a set distance out from the left. This is useful with printers which have few adjustments or when using a sheet feeder with some printers. The following command would set a left margin of 10 characters, for example.

```
LM   10
```

Page length

You should be clear that none of the adjustments mentioned above has any effect on page length. VIEW sets a standard length of 66 lines, unless you adjust it. This gives you 8 blank lines plus a header line at the top, and the same amount at the bottom, leaving 48 lines for text.

If you set the top, header, footer, and bottom margins all to zero and turned off headers and footers, you would have a blank header line at the top, a blank footer line at the bottom, and 64 lines for text in between.

In both cases the overall page length would be the same, but the page length can be changed. To set a page length of 70 lines, for example, enter the stored command PL in the stored command margin with 70 against it in the text area, like this:

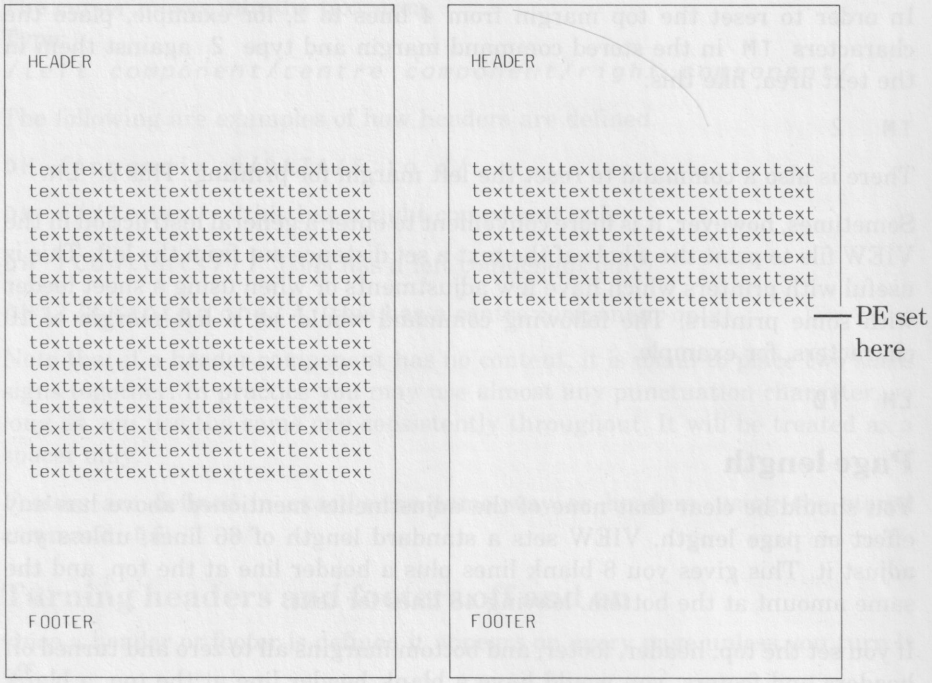
```
PL   70
```

It is particularly important to get the page length right if you are using continuous stationery. Measure your paper, try a few experiments once you have got VIEW printing, and set your page length accordingly.

Page end

So what happens when you enter PE in the command margin? – an immediate instruction to the printer to go to the top of the next page? Not quite.

What happens is much the same as if VIEW had come to the limit of the page length by itself: it adds the footer margin, the footer line and the bottom margin, and only then tells the printer to move to a new page.



So `PE` does not mean 'end the page immediately'. It means 'carry out the usual procedure for ending a page' – ie go to the bottom of the text area, add the footer margin, the footer line and the bottom margin, and only then move to a new page.

You may also use `PE` conditionally, to mean something like: 'end the page now if there is not room for the next paragraph'.

This is done by placing a number against the `PE` like this.

```
PE 10
```

This means that if there are fewer than 10 lines left out of the length set for the page, VIEW will perform a `PE` immediately – ie after allowing for the footer margin, footer line and bottom margin.

Suppose for example a page length of 68 has been set. The default vertical margins are unchanged at 4 lines each, taking up 9 lines (counting the header

line) at the top and another nine lines at the bottom. So the area available for text is 50 lines. By the time you reach line 43 of your text, therefore, you have only 7 lines left before VIEW reaches the end of its page length. So if you set `PE 8` at this point, this will cause VIEW to go into its page end routine immediately.

Summary of stored commands described so far

The following stored commands are available in VIEW. All are entered in the manner described at the beginning of this chapter.

To arrange text on the line

`CE` – Centres the text in relation to the left and right margin stops. If you are using side headings in the left margin area as described earlier, this command will ignore them and centre the text in relation to the margin stops alone. You can define an extra ruler so that the text is centred where you want it.

`RJ` – Right justify. Aligns the text with the right margin stop.

`LJ` – Left justify. Aligns the text with the left of the page. This can be used to protect text from formatting, since formatting is halted by a stored command.

Headers and footers

`DH` – Defines a header with left, centre and right components, in the form:

`DH /left component/centre component/right component/`

The slash signs are delimiters. To miss out a component, type two delimiters together.

`DF` – Defines a footer. The method is the same as for `DH`.

`HE` – Turns headers off or on. Use `HE OFF` or `HE ON`. When headers are off VIEW leaves a blank line when printing.

`FO` – Turns footers off or on. Use `FO OFF` or `FO ON`. When footers are off VIEW leaves a blank line when printing.

Margins

`TM` – Sets the top margin. Use `TM` *`number`*. If you do not set a top margin, VIEW assumes 4 lines.

`HM` – Sets the header margin. Use `HM` *`number`*. If you do not set a header margin, VIEW assumes 4 lines.

`FM` – Sets the footer margin. Use `FM` *`number`*. If you do not set a footer margin, VIEW assumes 4 lines.

`BM` – Sets the bottom margin. Use `BM` *`number`*. If you do not set a bottom margin, VIEW assumes 4 lines.

`LM` – Sets the left margin for printing purposes only. Use `LM` *`number`*. If you do not set a left margin, VIEW assumes zero.

Pages

`PE` – Tells VIEW to end the page, after having allowed for the vertical margins and the header and footer lines. If you end a page conditionally, use `PE` *`number`*. VIEW then ends the page if the number you supply is greater than the number of lines remaining on the page.

`PL` – Sets the page length, including vertical margins, headers and footers. Use `PL` *`number`*. If you do not set a page length, VIEW assumes 66 lines. Other stored commands are listed in the following chapters after their functions have been described.

12 Printing and numbering pages

Setting up VIEW for printing

VIEW operates most printers, serial or parallel. If a printer works with your computer it will work with VIEW.

For a general discussion of printing, read chapter 5. The instructions given here are in outline only.

The following operations are carried out on the command screen.

Load your printer driver with the instruction:

`PRINTER` *`name of driver`* **RETURN**

The name of the driver appears in the command screen heading and the driver is loaded. A printer driver is a program which operates the printer in conjunction with VIEW.

Set the computer for your printer in the usual way.

If in doubt, refer to your User Guide for details of using printers.

Set microspacing if you require it and if your printer and printer driver can use it. Type

`MICROSPACE` **RETURN**

The character `(m)` appears after the name of the printer driver in the command screen heading.

Microspacing evens up the spaces when a line of justified type is printed out, in order to improve its appearance.

You may also set the width of the characters in 1/120ths of an inch:

`MICROSPACE 15` **RETURN**

This would set the width to 8/120ths of an inch which is equivalent to 15 pitch. The default setting is 10/120ths, which is equivalent to 12 pitch.

Printing from memory or complete files

VIEW can print either from the text memory or from files on the filing system. To print from memory, switch to the command screen and type:

`PRINT` **RETURN** or `P` **RETURN**

To print a file, type

`PRINT` *`filename`* **RETURN** or
`P` *`filename`* **RETURN**

For example:

`PRINT agenda` **RETURN**

To print several files one after the other, type their names with a space between each. For example:

`PRINT agenda report letter` **RETURN**

It is worth noting that when arranged in this way the printing process literally follows on from one file to another. If there is no 'page end' instruction at the end of one file, the next file listed will start on the next line. This may be what you want, but if you want to separate your files, you should remember to insert `PE` commands at the end of each.

VIEW also remembers any instructions given it in the first file and applies these to the rest unless told to do otherwise. For example if you set a page length of 70 in the first file, VIEW will work to that measure for the rest of the files. You do not need to repeat the command at the start of each file.

Printing a page at a time

If you want to print a page at a time from memory or a file, you should use the command `SHEETS`, which sends a page at a time to the printer. Switch to the command screen and type:

`SHEET` **RETURN** (for text in memory)

`SHEET file1 file2` **RETURN** (for files on the filing system)

The message appears on the screen:

`Page 1..`

To print press the Space Bar.
To miss out a page press `M`.
To quit printing press `Q` or **ESCAPE**.

Printing unpaged

It is sometimes useful to be able to print out your text in a continuous column, with no page breaks in it, like a printer's galley.

You can do this with the stored command PB-PAGE BREAKS, like this:

```
PB  OFF
```

If you set this near the top of your text both the page ends that VIEW imposes at the ends of pages, and the page ends that you have entered yourself are no longer active.

To turn page ends on again, either delete the PB command or use

```
PB ON
```

Line spacing

VIEW normally prints on successive lines. If you require spaced lines, you can get them with the command LS-LINE SPACING, like this:

```
LS 1
```

The number against LS is the number of lines between each line of text. So the command above produces what is usually referred to as 'double spacing'.

To simulate printing on the screen

This is not strictly a printing command, and no printer driver or *FX commands are needed to make it work. Switch to the command screen and type

SCREEN **RETURN** (to display text in memory)

SCREEN file1 file2 file3 **RETURN**

The text appears a screenful at a time. To move on to the next screenful press **SHIFT**.

The SCREEN command is useful for making a rough check on the layout of the text, and checking where the page breaks occur. However, the printer driver will not be active, so bold and underlined type will not appear as such, and highlight codes will be displayed. Variations in line length may distort the appearance of the text.

Page numbers and number registers

You can set up VIEW to number the pages in your text for you. The method is to set up a header or footer in such a way that it prints the page numbers, incrementing by one as each page is printed.

The facility that makes this possible is VIEW's number registers. These are labelled A to Z, and all of them can be made to print out their values.

Register P counts the pages, and its value can be made to print in a footer like this:

```
DF  /Page :P///
```

This would print the current value of P as the left component of a footer. The vertical bar sign causes VIEW to interpret the P not as a character but as a number register value.

You can use this method in a single file or in a series of files. So if you define your footer as above in file Chap1, and enter the command

```
PRINT Chap1 Chap2 Chap3 Chap4 Chap5  RETURN
```

VIEW will continue to number your pages until it finishes Chap5.

The same applies to any commands set at the beginning of a series of files, as mentioned earlier in this chapter.

Setting registers

If you wish to number through from page 1 onwards you have only to make up your header or footer including register P. In practice, it is often necessary to start with a number other than 1, allowing for an unnumbered title page and other preliminaries.

You can arrange this by setting register P to the number you want.

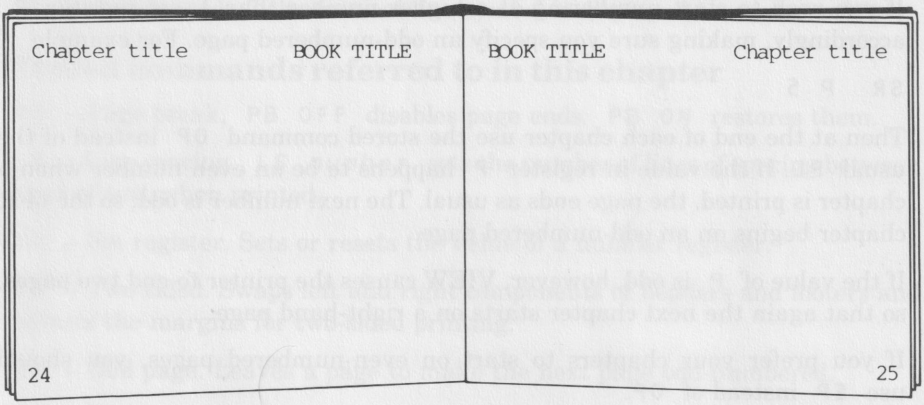
```
SR  P 5
```

would cause register P to start from 5. From then on, register P increments by 1 with every page end, providing automatic page numbering every time it is printed out as a footer.

Using page numbers in a two-sided layout

When processing books you may want a more formal page layout than is usual in a report. Page numbers, for example, normally appear on the outer edge of

the page, right on right pages and left on left pages. Book titles and chapter titles follow a similar pattern, like this:



You can achieve this effect in VIEW by setting headers and footers as usual, and then causing their right and left components to swap over when printing. To print pages as illustrated above, first set headers and footers as follows.

```
DH /Chapter title//BOOK TITLE/

DF /¦P///
```

`DH` and `DF` are of course stored commands, and are entered with EDIT COMMAND.

You then set up the stored command `TS` (two-sided) which swaps over the right and left components of headers and footers after each page end.

```
TS ON
```

To turn `TS` off again, replace `ON` by `OFF`.

An additional facility which is useful in book work enables you to increase the width of what printers call the 'gutter margin' – the two margins in the centre when a book is opened. These are normally made larger to allow room for the binding. So, for example, if you wanted to set these margins to 15 character spaces, you would modify the `TS` command to:

```
TS  ON 15
```

Starting chapters

In books and reports it is often useful to start all the chapters on a right-hand page – or maybe a left-hand page if that is your preference. The

normal PE-PAGE END can of course be used, but then you will have to count the pages. VIEW can do this automatically with a special page end command related to the number register P.

If you wish to start numbering at a higher number than 1, set register P accordingly, making sure you specify an odd-numbered page. For example

```
SR  P 5
```

Then at the end of each chapter use the stored command OP instead of the usual PE. If the value in register P happens to be an even number when a chapter is printed, the page ends as usual. The next number is odd, so the next chapter begins on an odd-numbered page.

If the value of P is odd, however, VIEW causes the printer to end two pages, so that again the next chapter starts on a right-hand page.

If you prefer your chapters to start on even-numbered pages, you should use EP instead of OP.

Other number registers

There are 26 number registers in all. P counts the pages and is incremented by 1 with every new page. L counts the lines and is reset to zero at every new page.

All registers can be set and reset with the SR command.

For example, suppose you want to print the chapter number as the header to each page, and you choose the register C for this purpose. You set C to 1 for the first chapter:

```
SR C 1
```

You then set up a standard heading for each chapter which increments C by 1 :

```
SR C !C+1
```

meaning 'set register C to C plus 1'. Your chapter heading is defined in the header:

```
DH  /Chapter !C///
```

which prints out as 'Chapter 1', 'Chapter 2', etc.

You may print out number register values by including their symbols (A to Z) in the following commands, preceded by the vertical line symbol.

DH – DEFINE HEADER
RJ – RIGHT JUSTIFY

DF – DEFINE FOOTER
LJ – LEFT JUSTIFY
CE – CENTRE TEXT

(Note that in mode 7 the vertical line symbol is displayed as ¦.)

Stored commands referred to in this chapter

PB – Page break, PB OFF disables page ends. PB ON restores them.

LS – Line spacing. LS *number* sets the number of lines of spacing between lines of text when printed.

SR – Set register. Sets or resets the value of a number register.

TS – Two sided. Swaps left and right components of headers and footers and adjusts the margins for two-sided printing.

OP – Odd page. Leaves a page to make the next page odd-numbered.

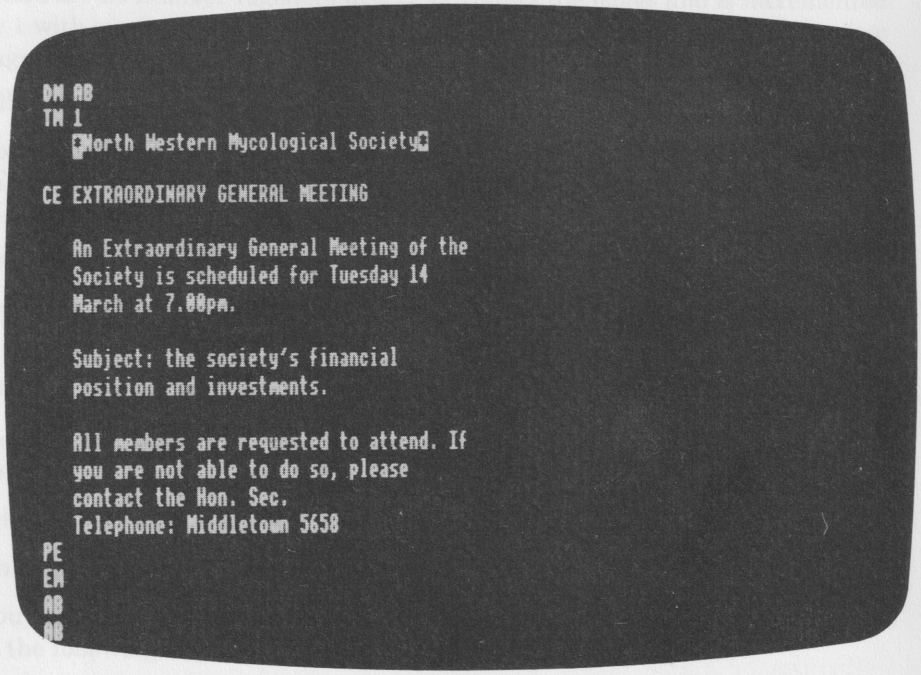
EP – Even page. Leaves a page to make the next page even-numbered.

13 Macros: for mailings and reports

VIEW's macro facility enables you to make up a collection of text and commands and to use it as often as you like with a minimum of trouble.

For example, if you are sending out essentially the same letter to a hundred people, with minor variations such as name and address, you can make the letter up into a macro. In the process of making it up, you give it a macro name. After that you can print the macro as often as you like, merely by entering its name in the margin as if it were a stored command.

The simplest form of macro is one which enables you merely to repeat the same text many times. The notice on the screen below shows how this is done.



The macro is that part of the text between the stored commands DM-DEFINE MACRO and EM-END MACRO. Beside the DM command is the macro's name: AB. The rest of the text in the macro is no different from any other VIEW text, with stored commands and highlights as usual.

So to cause the macro AB to be printed, place the cursor on a line of your choice and press EDIT COMMAND.
Type: AB **RETURN**
The AB remains in the stored command margin and is used like a stored command.

An important difference between a macro and other VIEW text is that a macro will not print by itself. It will only print in response to being 'called' by its name entered in the stored command margin. If you are keying in examples as you go through this book, try it out for yourself by typing SCREEN.

For example if you enter the macro name in the stored command margin ten times, followed by **RETURN** each time, the macro will PRINT or SCREEN ten times.

So to define a macro, the procedure is to enter the stored command DM on the line before the first text or command line. In the text area beside it place the name of the macro. This must consist of two characters only, which are not the name of any existing stored command. (If you name your macro PE for example, you will get a page eject when you try to use it.)

To end a macro, use the stored command EM.

To cause a macro to print, use the name of the macro as a stored command. Without this a macro cannot be printed or screened.

Macros with variations

The macro technique is particularly useful for mail shots or any other standard letters. The text of the letter is essentially always the same, but the name and address, and perhaps a few other details are varied according to the person to whom the letter is sent.

In order to introduce variations into the printing of a macro, it is necessary first to replace those words in the macro which are to be changed with one of the symbols: @0, @1, @2 ... @9.

When you enter the macro name as a stored command, the words to be included in the macro are typed beside the name.

So in our example above, the words EXTRAORDINARY GENERAL MEETING could be replaced with:

```
@0 GENERAL MEETING
```

Then when the macro is called, the line would read:

```
AB EXTRAORDINARY
AB ANNUAL
AB QUARTERLY
```

to cause VIEW to print `EXTRAORDINARY GENERAL MEETING`, `ANNUAL GENERAL MEETING` or `QUARTERLY GENERAL MEETING` as the case may be.

The words that fill in the blanks are known as parameters. In the case of an address, there will be several parameters to fill in. For example:

```
XY John Smith Esq.,25 Alpha Road,Middletown,Loamshire,Mr Smith
```

In the macro the address would appear like this:

```
DM  XY
    @1
    @2
    @3
    @4

    Dear @5,
```

and the printed result would be:

```
John Smith Esq.
25 Alpha Road
Middletown
Loamshire

Dear Mr Smith,
```

(If you try this out for yourself, do not forget to end your macro with the stored command `EM`.)

Notice that each item in the parameter line is divided off from the next by a comma. This could cause a problem if you wish to include a comma in a parameter. For example, you might wish to print the address in the example above: `25, Alpha Road`.

To avoid confusion between text commas and commas which serve merely to divide one parameter from the next, the method is to enclose any parameter that includes a comma within angle brackets. So the parameter line above would be changed to:

```
XY John Smith Esq.,<25,Alpha Road>,Middletown,Loamshire,Mr
```

As usual with VIEW commands, you should remember that VIEW interprets your instructions quite precisely. Notice that no spaces have been left after the commas in the example above. This may seem odd – you usually leave spaces after commas. But if you do so here VIEW will print exactly what you tell it to print – leaving an unwanted space before each of the items in the address.

Quite extensive use can be made of parameters in standard letters. Suppose we wish to send the following letter to applicants for a course, with variations to cover name, address, salutation, date, course, number, fee and department.

```
Mr A B Carter
10 Old Street
Newtown CX3 9JJ
15 Sept 1985

Dear Mr Carter,

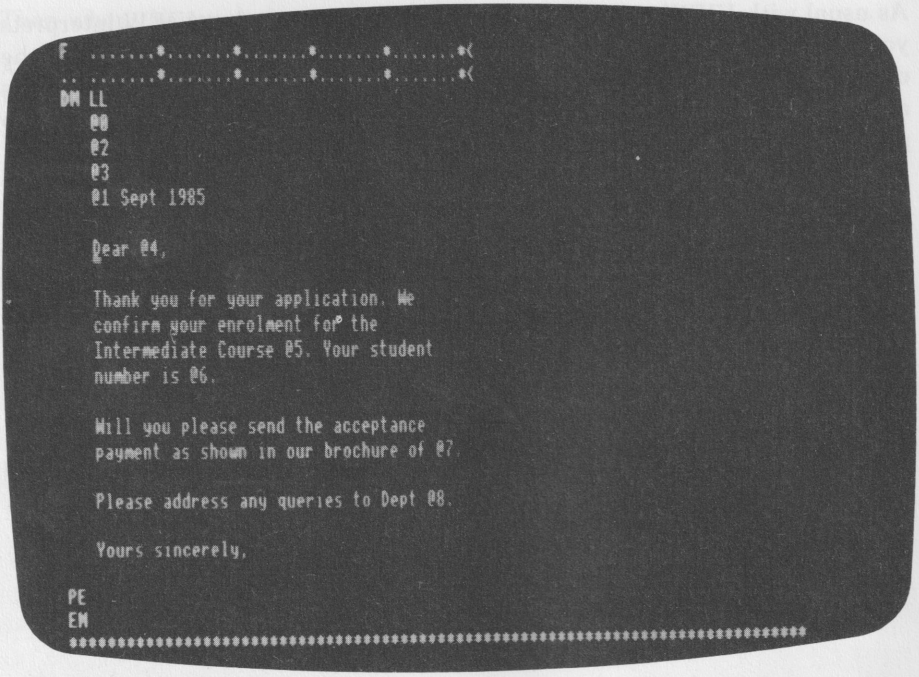
Thank you for your application. We
confirm your enrolment for the
Intermediate Course C. Your student
number is 552.

Will you please send the acceptance
payment as shown in our brochure
of £22.50.

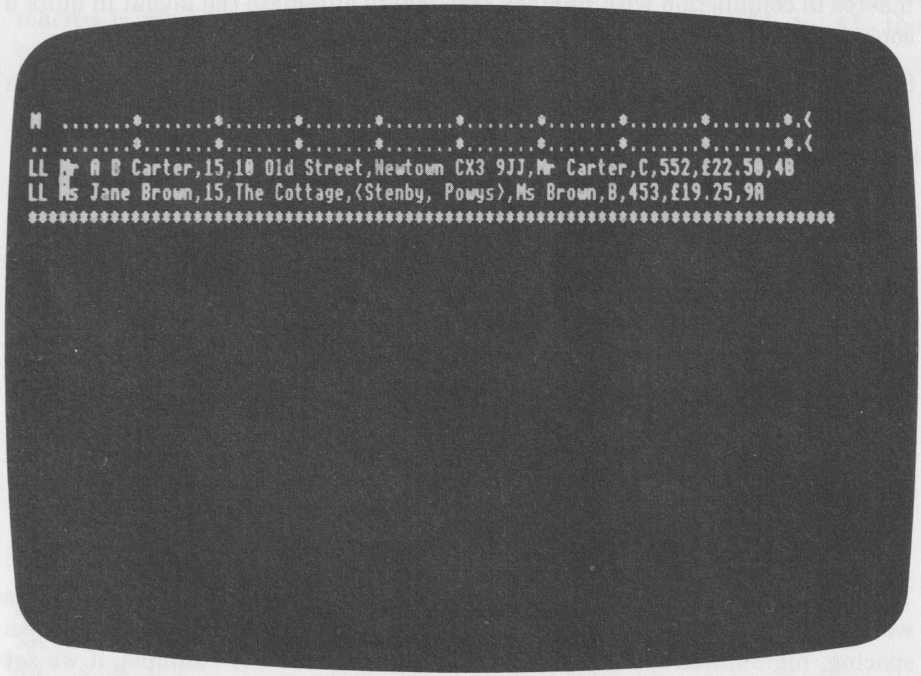
Please address any queries to Dept 4B.

Yours sincerely,
```

The equivalent macro is shown on the screen below – note how the variable items have all been replaced by 'blanks'.


F .....*.....*.....*.....*.....*(
.. ..*.....*.....*.....*.....*(
DM LL
@0
@2
@3
@1 Sept 1985
Dear @4,
Thank you for your application. We
confirm your enrolment for the
Intermediate Course @5. Your student
number is @6.
Will you please send the acceptance
payment as shown in our brochure of @7.
Please address any queries to Dept @8.
Yours sincerely,
PE
EM

Appropriate parameter lines would be as follows.



Note that since the macro `LL` is being used as a stored command, the `F` flag has changed to `M`.

If you are dealing with a very large mailing list, it would be inconvenient to have both macro and parameter lines in the same file, particularly since you will probably want to use the same names and addresses for subsequent letters and notices which you would make up into macros too.

In such cases you can record the macro in one file and the parameter lines in another. In this case we could record the letter in file `LETTER`, and the parameter lines in file `NAMES`. You would then be able to print out from the file with the command:

`PRINT letter names` **RETURN**

Try it out with a simulated print-out on the screen:

`SCREEN letter names` **RETURN**

Automatic layout

If you need to produce reports or books to a standard format, it is possible to use macros in conjunction with number registers to automate the layout in quite a sophisticated way.

Suppose you produce a series of reports, each with many chapters with headings and sub-headings, all numbered as follows.

```
CHAPTER 1

                    Chapter title

1.1 Section Heading

1.1.1 Sub-section heading
      text text text text text text text text text text text text text
      text text text text text text text text text text text text text
      text text text text text text text text text text text text text
      text text text text text text text text text text text text text

1.2 Section heading
      text text text text text text text text text text text text text
      text text text text text text text text text text text text text
      text text text text text text text text text text text text text
```

The most efficient way to manage a series of headings and paragraphs like this would be to set up a file containing macros to control all the standard features: spacing, highlighted type, and in particular numbers. For example, if we set register `C` to 1, and then cause a macro to increase register `C` by 1 each time a chapter heading is printed, we can print chapter headings completely automatically. Numbers for section and sub-section headings, and page numbers could be dealt with in the same way.

In the case of chapter headings you would set the register as follows:

```
SR  C 1
```

and each time a chapter heading was printed a macro would be called containing the line:

```
SR  C !C+1
```

which means that the new value of `C` is set to 1 greater than the old value of `C`. Later in the macro the value of `C` would be printed out by

```
LJ  *CHAPTER !C*
```

which uses the `LJ`-LEFT JUSTIFY command to print out the chapter heading, and the highlight 2 character to print it in bold. The vertical bar indicates that the `C` is a number register, not a character.

Using these principles we can construct a set of macros to manage a whole series of reports. The macros would be saved on a file, and used as part of the `PRINT` command whenever a report was printed. For example if a set of macros is given the name `BOOK` and the report files are `A1` to `A5` the command would be:

```
PRINT BOOK A1 A2 A3 A4 A5  RETURN
```

The macros below are an example of how this can be done. The way they do it may not be immediately apparent, but they will repay study.

Setting-up sequence

```
DF ///Page¦P/
SR  P Ø
SR  C Ø
SR  S Ø
SR  T Ø
```

(Define footers to display page numbers as register `P` which is automatically increased by 1 as each page is printed; also set chapter, section and sub-section heading numbers to zero.)

Chapter heading macro:

```
DM  CH                  (Define chapter heading macro.)
SR  C¦C+1               (Increase chapter number by 1.)
SR  S Ø                 (Set section number to 0.)
SR  T Ø                 (Set sub-section number to 0.)
PE                      (End page – so chapter begins on new page.)
LJ  *CHAPTER ¦C*        (Print chapter heading.)
CE  *@Ø*                (Print chapter title, bold, centred.)
EM
```

Section heading macro:

```
DM  SE                  (Define section heading macro.)
SR  S ¦S+1              (Increase section number by 1.)
SR  T Ø                 (Set sub-section number to 0.)
PE  5                   (End page if within five lines of bottom.)
LJ  *¦C.¦S @Ø*          (Print chapter and section numbers, and section
                         heading in bold.)
EM
```

Sub-section heading macro:

```
DM  SS                  (Define sub-section heading macro.)
SR  T ¦T+1              (Increase sub-section number by 1.)
```

```
PE  3                    (End page if within three lines of bottom.)
LJ  -¦C.¦s.¦T @Ø-        (Print chapter, section and sub-section numbers,
                          and sub-heading underlined.)
EM
```

When you write the text in VIEW, you have only to call the appropriate macros for chapter title, section heading, and sub-section heading, and to type the words of these headings opposite the macro names as parameters. VIEW does the rest, fitting your titles into the right place along with automatically generated numberings. The page numbers, of course, are supplied by the footers.

To use the method, the macros should be made into a separate file. For example, if the macros are made into a file called `MACRO` and the text into a file called `TEXT` the whole thing can be printed out with the command

`PRINT MACRO TEXT` **RETURN**

The screen below shows how the text is set, with no numbering or highlighting for headings. All that is provided by the macros.

```
F .....*.....*.....*.....*.....*.....*.....(
.. .....*.....*.....*.....*.....*.....*.....(


CH Aims and Scope

SE Organisation
   The Board and General Manager are located at
   Headquarters, and the five Regional Managers are at
   Regional Offices.
   Section Leaders report to Regional Managers.

SS Management
   The General Manager is responsible to the Board,
   Regional Managers to the General Manager, and Section
   Leaders to the Regional Managers.

SS Administration
   Reports must be filed regularly every week at Regional
   Offices and Regional Managers must submit summaries not
   later than the Tuesday of the following week.

SE Finance
   Since the Society is solely dependent on subscriptions
   and investments, every effort should be made to
```

When printed out the result is like this.

```
CHAPTER 1

                    Aims and Scope

1.1 Organisation
The Board and General Manager are located at Headquarters,
and the five Regional Managers are at regional Offices.
Section Leaders report to Regional Managers.

1.1.1 Management
The General Manager is responsible to the Board, Regional
Managers to the General Manager, and Section Leaders to the
Regional Managers.

1.1.2 Administration
Reports must be filed regularly every week at Regional
Offices and Regional Managers must submit summaries not
later than Tuesday of the following week.

1.2 Finance
Since the Society is solely dependent on subscriptions and
investments, every effort should be made to
```

If you make up files like this, and print them out, you will find that the first page printed is blank apart from the footer: `Page Ø`. The reason for this is that the chapter heading macro causes each chapter to be printed on a new page by doing a page end. The setting-up sequence sets the page number register `P` to zero, and the chapter heading macro then ends page 0. `P` is incremented to 1 as the page ends. This gives the correct starting numbers for both chapter and page: `Chapter 1 and Page 1.`

14 Using files

Naming files

To insert a filename into the heading on the command screen type

`NAME` *`filename`* **RETURN**

The file may then be saved to disc without naming it by typing

`SAVE` **RETURN**

The naming of the file has no effect on the text in memory. So, for example, you can save the contents of memory under one filename, edit and rearrange the contents, rename the file and save it again.

Saving text to a file

To save the whole contents of text memory to disc, type

`SAVE` *`filename`* **RETURN**

The filename may be in upper or lower case, or any mixture of cases. So `TEXT`, `text` and `Text` are equally valid.

To save the file named in the heading on the command screen type

`SAVE` **RETURN**

`SAVE` can be abbreviated to `SA`.

To save part of the contents of text memory only, first set markers 1 and 2 at the beginning and end of the part you wish to record. Then type

`WRITE` *`filename`* `1 2` **RETURN**

The `WRITE` command without markers has the same effect as `SAVE` except that it is slower. `WRITE` can be abbreviated to `W`.

Retrieving text from a file

To clear the text memory and load the whole contents of a file type

`LOAD` *`filename`* **RETURN**

The name of the file appears in the command screen heading. `LOAD` can be abbreviated to `L`.

To preserve text in memory and add the contents of a file to the end of text in memory in so far as memory allows, type

`READ` *`filename`* **RETURN**

The command screen heading is unchanged. If there is not enough memory available to read the whole of the file, VIEW gives you the message: `Not all read in.`

To read the contents of the file into a specific place in the text memory, first place marker 1 at the point where you want the text read in.
Then type

`READ` *`filename 1`* **RETURN**

Editing BASIC programs

VIEW allows you to read programs written in BASIC into memory. This can be convenient both for editing BASIC programs, and if you want to quote from BASIC programs within the text you are writing. The one restriction is that the BASIC program may not have lines longer than 132 characters.

To place the BASIC program in a file which VIEW can read

Type: `*BASIC` **RETURN** (to get into BASIC)
Type: `LOAD` *`"FILENAME"`* **RETURN** (to load the BASIC program)
Type: `*SPOOL` *`NEWFILE`* **RETURN** (to create a new file)
Type: `LIST` **RETURN** (to get the program into the new file)
Type: `*SPOOL` **RETURN** (to close the new file)

To read *`NEWFILE`* into VIEW for editing

Type: `*WORD` **RETURN** (to get into VIEW)
Type: `READ` *`newfile`* **RETURN** (DO NOT USE LOAD SINCE THIS IS NOT A VIEW FILE)

To use the edited version as a BASIC program

Type: `*BASIC` **RETURN**
Type: `*EXEC` *`NEWFILE`* **RETURN**

The program is now back in memory and can be listed and run.

15 Continuous processing

Until now we have described VIEW in terms of composing text, editing and saving it, loading and modifying, perhaps reading files into it, and printing it out. All the processing is concentrated on that single column of text on the text screen.

Many people always use VIEW in this way, and find it very satisfactory. But there is another possibility which has advantages with large amounts of text.

This is the `EDIT` facility, in which you can read in some of the text from a file, process it, and send it on to another file, while reading more in from the first file. The process continues until you have passed all the text through to the second file.

How to EDIT

To start the `EDIT` procedure, by reading in your file, and giving a name to the file to which the text will go after you have edited it type:

`EDIT` *`file-in file-out`* **RETURN**

(*`file-in`* is your original file, and *`file-out`* the destination file.)

VIEW now reads text from *`file-in`*. When you have edited it and are ready to go on to the next batch, type

`MORE` **RETURN**

The text you have edited is now written in to your *`file-out`* and new text is read into memory from *`file-in`*.

When the last batch of text is passed through to *`file-out`* the process is finished.

To stop EDITing

If you wish to stop editing before you have finished the document, the normal way is to type

`FINISH` **RETURN**

The text in memory is placed in *`file-out`* and any text that has not yet been read from *`file-in`* is read and transferred directly to *`file-out`*.

You are now left with two files: the original file (`file-in`) and the destination file (`file-out`). If you no longer need `file-in` delete it.

When you next take up the task of editing the text, your old *`file-out`* will become your new *`file-in`*. All you have to do is to keep on asking for `MORE` until you come to the point where you left off editing last time.

To abandon EDITing

If you have scarcely started editing before you have to abandon the task you may feel it is not worthwhile to have all the text transferred to *`file-out`*. In such cases you should type

`QUIT` **RETURN**

which simply stops the process, leaving *`file-in`* intact, but *`file-out`* incomplete. You should then delete *`file-out`*.

Naming files

The normal rules for naming files apply to `EDIT` – see the Reference chapter.

You should be particularly careful in your choice of names when using `EDIT`. When working at speed it is easy enough to become confused with normal saving and loading. This applies even more when you are handling two files at the same time.

Transferring text

Most of the time when using `EDIT` you will rely on the `MORE` command to feed through the next batch of text.

Sometimes, however, you may wish to transfer part of the text you are working on to *`file-out`* in order to get some more out of *`file-in`*, perhaps to compare or move parts of it.

You can do this by setting marker 1 at the point up to which you want text to be transferred to *`file-out`*. Then type

`MORE 1` **RETURN**

The first part of the text will then go to *`file-out`* and more material will arrive from *`file-in`*.

16 Notes

Count

To find out the number of words in the text memory, switch to the command screen and type

`COUNT` **RETURN**

If you want to know the number of words in part of the text only, set markers 1 and 2 before and after the part concerned and type

`COUNT 1 2` **RETURN**

While this is a useful guide it is important to realise what VIEW is doing when it counts words. It would be truer to say it is counting the spaces between the words rather than the words themselves. Since VIEW cannot read the only way it can detect a word is by the fact that words have spaces each side of them. so if you have type `WORD` it will recognise it as one word. If you have typed `W O R D` VIEW will count it as four words. VIEW will not count the words on a stored command line except in the case of the stored commands `CE`, `RJ` and `LJ`.

Memory

We have referred to memory several times in this book. Unless you are using shadow memory, the amount of memory available for text varies with the screen mode you are in.

If you wish to process large files one option is described in the last chapter – the `EDIT` procedure.

Memory Full – Press ESCAPE

If you compose so much text that you fill up text memory, the computer will bleep and you will see the message above in white at the top of the screen. When you press **ESCAPE** you will see that there are very few bytes left.

If you are in mode 3 without shadow memory, you have the option of changing to mode 7 and continuing to compose text, or of splitting the text into two files.

To split the text into two files, first `SAVE` the text as it is as *`filename`*. Then place markers, one in the middle of the column of text and the other at the end. Delete the end part with DELETE BLOCK and save the top part as

filename1. Load the whole file again. Delete the top part and save the bottom as *filename2*. Since you no longer need *filename* you can delete it.

SETUP

The usual way of changing the FJI flags at top left in screen mode is to press the appropriate function key, for example JUSTIFICATION.

You can also change them in the command screen with the SETUP command. So, for example, if you want formatting on, no justification, and insertion of text you would type

SETUP FI **RETURN**

You can abbreviate SETUP to SET.

17 Abbreviations

You may use the following abbreviations in the command screen.

CHANGE	C
CLEAR	CL
COUNT	CO
EDIT	E
FIELD	FI
FINISH	F
FOLD	FO
FORMAT	FOR
LOAD	L
MICROSPACE	MI
MODE	M
MORE	MO
NAME	N
NEW	NEW
PRINT	P
PRINTER	PRINTE
QUIT	QUIT
READ	RE
REPLACE	R
SAVE	SA
SCREEN	SC
SEARCH	S
SETUP	SET
SHEETS	SH
WRITE	W

The '*' symbol introduces a filing-system command. For details see the appropriate *Filing System User Guide*.

18 Reference

Immediate commands

These commands are issued from the text screen by using the function keys which are at the top of the keyboard. Some commands are given by pressing the function key alone. Other commands require **SHIFT** or **CTRL** to be pressed at the same time as the function key. Immediate commands are listed on the function key card.

FORMAT PARAGRAPH

Causes formatting to take place from the cursor line to the end of the paragraph. The end of a paragraph is defined as any of the following: a blank line; a line with tabs on it other than the left margin tab character; a line beginning with a space; or, if the left margin is active, a marginal note. A tab in the left margin will act as a marginal note and hence formatting will stop at this point. A space character in the left margin will not have this effect. A paragraph can be protected from formatting by having a ruler above it which has no right margin stop set. Note that formatting is active only when the F flag is displayed at the top left of the screen. Formatting is active by default and can be turned off and on by pressing FORMATTING.

See also: FORMATTING, JUSTIFICATION

TOP OF TEXT

Takes the cursor to the first character of the text in memory. **CTRL** with the ↑ key has the same effect.

BOTTOM OF TEXT

Takes the cursor to the last character of the text in memory. **CTRL** with the ↓ key has the same effect.

DELETE END OF LINE

Deletes the character the cursor is on and the rest of the cursor line to the right of the cursor.

BEGINNING OF LINE

Takes the cursor to the first character of the cursor line within the main text area. **CTRL** with the ← key has the same effect.

END OF LINE

Takes the cursor to the last character of the cursor line within the main text area. CTRL with the → key has the same effect.

INSERT LINE

Inserts a line above the cursor line. The text below the new line moves down to accommodate the new line.

DELETE LINE

Deletes the cursor line. The text below the deleted line closes up.

INSERT CHARACTER

Inserts a space before the character on which the cursor is placed by moving the text from the cursor position one character to the right.

DELETE CHARACTER

Deletes the character on which the cursor is placed and closes up the text to the right.

MOVE BLOCK

Moves a marked block of text to the cursor position. If the cursor is in the marked block or on marker 2, the computer will bleep to signify an error and nothing will happen. If there is insufficient memory available, the block will not move. This is because MOVE BLOCK first makes a copy of the marked block at the cursor position and then deletes the original block, so there needs to be sufficient room in memory for two blocks.

See also: SET MARKER

SWAP CASE

Changes the case of the letter on which the cursor is placed and moves the cursor one place to the right.

MARGINS

Moves the cursor to the beginning of the main text area if the cursor is in the left margin area and to the beginning of the left margin area if the cursor is in the main text area.

DELETE UP TO CHARACTER

Deletes from the cursor position up to and including the first occurrence of the character specified on the cursor line. If there is no occurrence of the character

specified on the cursor line, the computer will bleep. A good way to delete a word is to place the cursor on the first character of the word and press DELETE UP TO CHARACTER followed by the Space Bar. The word will then be deleted. If the character you have specified appears more than once consecutively, then each consecutive occurrence of it will be deleted. This specifically caters for deleting a word at a time since there may be more than one space between words.

HIGHLIGHT 1 and HIGHLIGHT 2

Used to specify printing effects, such as bold, underlined, or italic type, and extended characters; see p.125. In modes 0 to 6 the highlight characters appear as a reversed out minus sign and asterisk respectively. In mode 7 they appear as a minus sign and asterisk respectively. The highlight 1 character represents the code 128 and highlight 2 the code 129. These are translated by the printer driver into the appropriate printer codes.

Note that the highlight characters are ignored for justifying. Therefore, if their insertion affects the layout of your text on the screen, this will not be reproduced on printing.

GO TO MARKER number (1 to 6)

Takes the cursor to the marker specified. This applies to markers 1 to 6. If the marker specified is not set, the computer bleeps to signify an error and nothing happens.

See also: SET MARKER

SET MARKER number (1 to 6)

Sets a marker at the cursor position with the number specified. Markers 1 and 2 are shown as blocks of reverse video. Markers 3 to 6 are not visible. A block of text can be placed between markers for copying, deleting or moving. Press SET MARKER followed by 1 with the cursor on the first character to be included in the block of text. Then move the cursor to after the last character to be included in the block and press SET MARKER followed by 2. To clear markers, return to command mode and type `CLEAR` . Markers are also cleared when text containing them is formatted or when a block of text is moved. When text is saved to a file markers are not saved with it. If text containing a marker is deleted then the marker is also deleted.

See also: MOVE BLOCK, DELETE BLOCK, GO TO MARKER

EDIT COMMAND

Moves the cursor into the stored command margin, allowing a stored command

to be entered or altered. Pressing **RETURN** or **ESCAPE** takes the cursor out of the stored command margin allowing the entry of text.

See also: DELETE COMMAND

DELETE COMMAND

Deletes a stored command on the cursor line provided that the cursor is not in the stored command margin.

See also: EDIT COMMAND

DELETE BLOCK

Deletes a marked block of text. This only applies to a block marked by markers 1 and 2. If either of the markers has not been set, the computer bleeps and does nothing.

See also: SET MARKER

NEXT MATCH

Finds the next occurrence of the string specified in the `SEARCH` command after the previous occurrence of that string. Once there are no further occurrences VIEW returns to the command screen.

FORMATTING

When formatting is active an `F` is displayed in the top left hand corner of the screen. Formatting is active by default. Whole words are transferred to the next line if they would otherwise go beyond the right margin stop and there is no need to press **RETURN** at the end of each line.

See also: FORMAT PARAGRAPH, JUSTIFICATION

JUSTIFICATION

When both formatting and justification are active `FJ` is displayed in the top left hand corner. Once again, there is no need to press **RETURN** at the end of each line, but in this case, the text will also be justified; text will be aligned under the right margin stop. Justification can be switched on either before typing the text or afterwards in which case the text will be justified as a result of pressing FORMAT PARAGRAPH or typing `FORMAT` in the command screen.

See also: FORMATTING, FORMAT PARAGRAPH

INSERT/OVERTYPE

When inserting is active, I is displayed in the top left hand corner of the screen. Inserting is turned on and off by pressing INSERT/OVERTYPE. The existing text moves right to accommodate the new text you type in. This is in contrast to the default overtyping state in which new text will replace the existing text. Having inserted new text you may need to press FORMAT PARAGRAPH to tidy up the text. The **DELETE** key operates differently when inserting is active. Rather than simply deleting the character before the cursor, leaving a space, the text after the cursor is closed up.

RULER

Inserts a line above the cursor line and puts on it the default ruler for the current screen mode. This is in contrast to **SHIFT COPY** which puts a copy of the current ruler on to the line.

SPLIT LINE

Splits the cursor line at the cursor position, moving the right hand part of the line on to the line immediately below the cursor line. All following text moves down to accommodate the new line. Take care to put the cursor on the first character you want to be moved on to the line below. If you place the cursor on a space before the first character you want to move, then the space will be moved to the beginning of the new line which means that, when you press FORMAT PARAGRAPH to tidy up the text, formatting will be prevented by the leading space.

JOIN LINES

Joins the line below the cursor line to the end of the cursor line. The cursor may be anywhere on the upper of the two lines. Be careful of cases where the line is too long. The line will be split at the word boundary nearest to the 132 character position.

MARK AS RULER

Puts two dots in the stored command margin of the cursor line. You can make up your own ruler on the line using dots, margin stops, tab stops, and b characters.

Other text screen commands

These are entered by using keys other than the function keys.

DELETE

Deletes the character to the left of the cursor and moves the cursor a space to the left. When inserting is active the text closes up.

ESCAPE

Switches between the command screen and the text screen and can also be used to cancel commands which require parameters before entering the parameter(s). Such commands include DELETE UP TO CHARACTER, SET MARKER, EDIT COMMAND, GO TO MARKER and `SEARCH`. With the cursor in the stored command margin, pressing **ESCAPE** takes the cursor out of the stored command margin. Pressing **ESCAPE** is also the response to the `Memory full` error message.

COPY

Makes a copy of a marked block at the cursor position. The original block remains marked. There must be sufficient room in memory for the two copies of the block. If markers have not been set, the computer bleeps to signify an error and nothing happens.

SHIFT COPY

Inserts a copy of the current ruler above the cursor line.

BREAK

Resets the machine and returns you to the command screen. Pressing **BREAK** is not recommended.

CTRL BREAK

Completely resets the machine and may lose the text in memory.

Programming function keys

In addition to the above, the function keys can be programmed in the command screen to produce a particular string in the text screen. For example, to program the function key **f0** to produce the string `computer` enter the command screen and type:

```
*FX 228,1 RETURN
*KEY 0 computer RETURN
```

In the text screen, holding down both the **SHIFT** key and the **CTRL** key while pressing **f0** will produce the word `computer` at the cursor position. Any function key can be programmed in the same way.

Stored commands

These are used in the text screen. Press EDIT COMMAND to move the cursor into the stored command margin, type the code and press **RETURN** followed by any necessary parameters. In the examples given below, n is a number.

■ `CE` *`text`*

Centre. Centres the text between the right and left margins. With a left margin active, text is centred in relation to the main body of the text which lies between the margin stops.

■ `RJ` *`text`*

Right Justify. Aligns the text to the right margin. Note that tabs will be reduced to single spaces.

■ `LJ` *`text`*

Left Justify. Aligns the text to the left of the page. This is particularly useful for protecting addresses from formatting since formatting is halted by a stored command. Each line of the address can be preceded by the `LJ` stored command and thus will not be affected by formatting.

■ `DH` */left/centre/right/*

Define Header. Defines the page header with left justified, centred and right justified components or any combination of these specified between delimiters. The delimiters can be any of the following characters: `! " # $ % & ' * + , = . /` The delimiters must always be present even where one component is not defined. If no page header is defined a blank line is printed nevertheless.

■ `DF` */left/centre/right/*

Define Footer. Defines the page footer with left justified, centred and right justified components or any combination of these specified between delimiters which may be any of the following characters: `! " # $ % & ' * + , - . /` The delimiters must always be present even if one of the components is not defined.

■ `HE ON/OFF`

Header. `HE OFF` turns the page header off which means that the currently defined page header text is not printed out until the page header is turned on again using `HE ON`. However, a blank line is always printed when the header is turned off to maintain page layout ie `HE OFF` means that a blank line will be printed. `1` and `Ø` can be used instead of `ON` and `OFF`.

■ FO ON/OFF

Footer. FO OFF turns the page footer off which means that the currently defined page footer text is not printed out until the page footer is turned on again using FO ON. However, a blank line will be printed when the footer is turned off to maintain page layout ie FO OFF means that a blank line will be printed. 1 and Ø can be used instead of ON and OFF.

■ DM *1 or 2 letter name*

Define Macro. You must give the macro a one or two letter name which must be upper case. The name must not be the same as any of the pre-defined stored command names. If you do use a stored command name, the stored command will be called rather than the macro.

■ EM

End Macro. You must use this stored command at the end of your macro.

■ SR *letter expression*

Set Register. Sets the register specified by the letter to the value of the expression. The expression may include references to other number registers and the + and - operators. For example,

```
SR P ¦R+1
DF //Page¦P//
```

set register P to an initial value of register R plus 1 and then defines the page footer to print the page number stored in register P.

■ PB ON/OFF

Page Break. PB ON enables page breaks. PB OFF disables page breaks. VIEW has page breaks enabled by default.

■ PL *n*

Page Length. Sets the page length to *n* lines. The default page length is 66 lines. This includes headers, footers, top margin and bottom margin. The page length can be set from 3 lines to 255 lines.

■ TM *n*

Top Margin. Sets the top margin to *n* lines. Thus TM Ø turns off the top margin. Defaults to 4 lines.

■ HM *n*

Header Margin. Sets the header margin to *n* lines. Thus HM Ø turns the header margin off. Defaults to 4 lines.

■ FM *n*

Footer Margin. Sets the footer margin to *n* lines. Thus FM Ø turns the footer margin off. The default is 4 lines.

■ BM *n*

Bottom Margin. Sets the bottom margin to *n* lines. Thus BM Ø turns the bottom margin off. The default is 4 lines.

■ LM *n*

Left Margin. Sets the left margin to *n* spaces. This is used to position text on the printer. The default is 0 spaces.

■ LS *n*

Line Spacing. Causes *n* blank lines to be printed between each line of text.

■ TS ON/OFF *n*

Two Sided. TS OFF disables two sided printing. TS ON enables two sided printing, and if n is supplied *n* extra spaces are printed in the left margin of odd numbered pages. The page header and page footer are reversed on even numbered pages.

■ PE *n*

Page End. PE with no number after it means carry out the usual procedure for ending a page ie go to the bottom of the text area, add the footer margin, the footer line and the bottom margin, and then go to a new page. If a number *n* is supplied, then VIEW carries out the page end if the number specified is greater than the number of lines remaining on the page.

■ OP

Odd Page. Gives one page end if on an even numbered page, two page ends otherwise.

■ EP

Even Page. Gives one page end if on an odd numbered page, two page ends otherwise.

■ HT *character n*

Highlight. Sets the specified highlight character to the number *n*. The character will either be - or *. For example, HT * 13Ø sets highlight 2 to 130. 1 and 2 can be used for - and * respectively.

Command screen commands

These commands are used in the command screen. The minimum abbreviation is shown at the extreme right-hand side of each command. Type in the command and the syntax which follows it where necessary then press **RETURN**.

■ `*BASIC` `*B.`

Switches the computer into BASIC. Note that you should save any required text in memory to disc before using this command as otherwise it will be lost since this command clears the memory.

■ `CHANGE` *`target result`** `C`

Finds all occurrences of the target string and replaces them with the result string. In the case of the `CHANGE` command this change happens automatically. It is often wiser to use the `REPLACE` command (see below) which acts in the same way except that you choose whether to change each occurrence of the target string or not. There is thus far less chance of making mistakes with the `REPLACE` command.

If changing one word for another word, you must use delimiters other than spaces so that spaces can be used to isolate the word; this prevents the word also being changed when it forms part of a longer word. For example an attempt to change `cat` to `dog` could result in `catalogue` being changed into `dogalogue`. The syntax

```
CHANGE/ cat / dog /
```

prevents this from happening; only the word `cat` will be changed into `dog`. With the `REPLACE` command you would not have this problem since if asked whether you wanted to change `catalogue` you would simply type `N`.

When changing a single word which you are absolutely certain only occurs on its own and not as part of another word, spaces suffice as delimiters. For example,

```
CHANGE ladder rope
```

When changing a string for another string, delimiters other than spaces are necessary. For example,

```
CHANGE/Peter Rabbit/Jemima Puddleduck/
```

This has slashes as delimiters since VIEW takes the first character after the target string as the delimiter and would assume that the space after `Peter` was the delimiter if no other delimiting character were present.

If you are not sure of the spelling of the target string or if it is not made up of printable characters there are special characters you can use. These are also used with the REPLACE command and with the SEARCH command. They are as follows:

^C	Carriage Return
^S	Hard space
^Z	Soft space
^T	Tab
^L	Left margin tab
^^	^ character
^-	Highlight 1
^*	Highlight 2
^?	Any character

For example, to change all HIGHLIGHT 1 characters to HIGHLIGHT 2 characters, you would type:

```
CHANGE ^- ^*
```

■ CLEAR CL

Removes all markers from the text.

■ COUNT* CO

Counts the number of words in memory or between the specified markers. The total number of words is actually arrived at by counting the number of gaps between words. Thus if you have used double spacing between the letters of headings for exmaple, this command could produce a misleading result.

■ EDIT *filein fileout* E

Starts editing a file without the restriction that it must fit into the available memory. Reads in the first part of *filein* and edits the text passing it out to *fileout*. When all the text in memory has been written to *fileout*, MORE reads more text from *filein* which is then edited and passed to *fileout*. FINISH closes both files, ends the EDIT session and reads all the text from *filein* to *fileout*. QUIT does the same but does not read all the text from *filein* to *fileout*. At the end of the EDIT session, *filein* can then be deleted and *fileout* kept as the edited version.

■ FIELD *ASCII number* FI

Assigns the tab function to the key which has the specified ASCII number. Reset to the default value with FIELD 9.

■ FINISH F

Finishes an EDIT session. Both files are closed and the rest of *filein* is written to *fileout*.

■ FOLD ON/OFF

Turns the facility to ignore case on and off with SEARCH, CHANGE and REPLACE. When folding is on, which it is by default, VIEW replicates the case in the result string of corresponding characters in the target string. Thus capital letters are replaced by capital letters and lower case letters by lower case letters. For example, with folding on, typing:

```
CHANGE cat owl
```

would result in cat being changed to owl, CAT to OWL and Cat to Owl. With folding off, case is ignored so that typing:

```
CHANGE basic BASIC
```

would find only basic, not BASIC or Basic and would force the change to BASIC.

To find out whether folding is on or off, type FOLD **RETURN**. VIEW will reply with the message Folding off or Folding on.

■ FORMAT* FOR

Formats all the text in memory or between specified markers.

■ LOAD *filename* L

Loads a file from the filing system into memory assuming it fits into memory. This file replaces whatever was in memory before. If using cassettes, you should use the READ command instead.

■ MICROSPACE *n* MI

Enables microspacing assuming this is possible with the current printer driver. Optional *n* is the width of the characters in 1/20ths of an inch. The default value is 10 which is equivalent to 12 pitch. VIEW will check that the current printer driver supports microspacing and if so (m) will be displayed after the printer driver name in the command screen. VIEW microspaces by dividing evenly between the words the soft spaces that were inserted when a line was formatted. No microspacing will occur if the line contains tabs so that tables and columns set out with tabs will remain perfectly aligned.

■ MODE *n* M

Switches the computer into screen mode *n*. Screen modes in VIEW give you 14, 34, or 74 characters across the screen and variable amounts of text space.

Experiment with the modes until you find the one with which you are happiest. Note that if you have shadow memory fitted, all modes have an equal amount of text space available. Refer to your shadow memory documentation for details on changing modes.

The ruler you have when first entering the text screen is determined by the mode you were in on entering VIEW. To get a default ruler for the mode you are using press RULER. For example, when you enter VIEW by typing `*WORD` you may be in mode 7. If you then type `MODE 3` **RETURN** and press **ESCAPE** you will be in the text screen in mode 3 but the ruler at the top of the screen will be the mode 7 ruler because that is the mode you were in on entering VIEW. To get the mode 3 ruler press RULER. It is good practice when starting documents to put a ruler at the top so that VIEW always formats to your specified ruler rather than to its own current ruler.

■ `MORE*`

During an `EDIT` session, writes text to *`fileout`* and reads more text from *`filein`* into memory. You can use `MORE` to join two text files (`file1` and `file2`) each of which fits into memory but whose combined length is too long. They then form one large file (`bigfile`). Type:

`EDIT` *`file1 bigfile`* **RETURN**

then set marker 1 at the end of `file1` and in the command screen, type:

`MORE 1` **RETURN**
`READ` *`file2`* **RETURN**
`FINISH` **RETURN**

`Bigfile` will now be the two smaller files joined together.

■ `NAME` *`filename`* N

Names or renames the text in memory. Having given the text a name you can then save it simply by typing `SAVE`; it will be saved with the name you have given it.

■ `NEW`

Prepares VIEW for typing a new document. It clears the text in memory and resets the `Editing No File` line.

■ `PRINT` *`file1 file2 ...`* P

Prints the file or files specified on to continuous stationery. If you are printing the text which is in memory you do not need to give the filename; simply type `PRINT`. If you are printing one or more files you must specify the filename(s) in the order in which you wish them to appear. If you are printing several files and you have used a number of stored commands such

as PL, HM, etc you need only use these commands at the beginning of the first file or as the first file; you do not need to repeat the commands at the top of every file. Nor is it necessary to use the PE command other than in cases where a page break would be awkward such as in the middle of a table. Otherwise, VIEW automatically performs page breaks to the page length specified. VIEW's printer driver is active by default. If you want to take advantage of the particular facilities of your printer you will need to load a printer driver. The *Printer Driver Generator* package provides standard printer drivers and in addition allows you to define your own. The PRINT command works by sending text to the printer driver. Pressing **ESCAPE** stops any more text being sent to the printer driver but printing will not necessarily stop instantly since the printer buffer may be sending the last of the previous batch of text to the printer. Note that there should be no line spaces left between the current ruler and the stored commands which govern printing.

■ PRINTER *filename* PRINTE

VIEW is able to drive any printer which can work with the BBC Microcomputer. Incorporated in VIEW is a default printer driver which drives any parallel or serial printer with no highlighting supported. The PRINTER command allows you to load a specified printer driver into VIEW's driver area.

When a new printer driver has been loaded this driver will be used for all printer output until the PRINTER command is used again. To revert to the default printer driver type

PRINTER

with no filename.

■ QUIT

Abandons an EDIT session. Both *filein* and *fileout* will be closed but the text in memory and any text remaining in *filein* will not be written to *fileout*.

■ READ *filename** RE

Reads a file into the end of the text currently in memory or to a marked place in that text. If there is not enough memory for all the text to be read in, as much as possible will be read in. The READ command gets rid of any undesirable characters. If the lines in the file to be read in are longer than 132 characters, Carriage Returns are inserted. This means that you can read non-VIEW files into VIEW. Spooled ViewSheet files can be read into VIEW.

■ REPLACE *target result** R

Finds the first occurrence of the target string and gives the user the option whether to replace it with the result string or not. The same option is offered for

each subsequent occurrence of the string. If you press N, no change takes place. If you press Y, the target string is replaced by the result string. This is sometimes preferable to CHANGE which offers no option.

The same special characters can be used with REPLACE as with CHANGE.

See FOLD above.

■ SAVE *filename* SA

Saves the text in memory to the filing system with the name specified. Any file already saved under that name will be replaced. If you have already given the text a name using the NAME command or have loaded a file you can use simply SAVE.

■ SCREEN *file1 file2 ...* SC

Displays the text on the screen as it will appear when printed out. If you simply type SCREEN the text in memory will be displayed. To display files from the filing system you must specify the filename or filenames. The text is displayed a screenful at a time. To see the next screenful, press and release **SHIFT**.

■ SEARCH *string** S

Searches the text for the first occurrence of the string specified. Press NEXT MATCH to find subsequent occurrences of the string.

The same special characters can be used with SEARCH as with REPLACE and CHANGE.

See FOLD above.

■ SETUP FJI SET

Sets any or all of the text mode flags. SETUP alone turns all the flags off.

■ SHEETS *file1 file2 ...* SH

Prints the text pausing at the end of each page to allow the user to feed in the next sheet of paper. If you want to print out the text in memory in this way, you only need to type SHEETS. If you are referring to one or more files you need to give the filenames. After you type SHEETS and press **RETURN** you will see Page 1. . Press the Space Bar to begin printing. At the start of the second page, you will see Page 2. . and so on. To stop text being sent to the printer driver press M to miss out a page or **ESCAPE** to quit printing.

■ WRITE *filename** W

Writes the text out to a file. This is slower than SAVE but has the advantage that it can be used with markers.

* These commands can be used with markers

Filenames

VIEW stores the current filename in memory, and allows up to 19 characters.

*EXEC files

A `*EXEC` file is a text file containing commands which would normally be given from the command screen. This is time saving when there are a lot of commands which you frequently enter at the same time. For example, to create a `*EXEC` file containing the commands to switch to VIEW, change to mode 3, prepare for typing a new document, switch justification off and inserting on and to give your text file the name `TEMP`, you would type, in the text screen

```
*WORD
MODE 3
NEW
SET FI
NAME TEMP
```

then return to the command screen and type `SAVE` *`filename`*. Then instead of having to type in all the commands in the command screen, you would type simply `*EXEC` *`filename`*. Thus you save all the commands in a text file and then `*EXEC` the file when necessary.

The format of VIEW files

VIEW files are made up of lines of text of up to 132 characters followed by a Carriage Return. Stored commands are represented by the character `&80` followed by the two characters of the macro call or stored command followed by the text. Ruler lines are represented by `&81` followed by `..` (two dots). Tab characters are represented by `&9`. Left margin tab characters are represented by `&B`. Carriage Return is represented by `&D`. Soft spaces are represented by `&1A`. Hard spaces are represented by `&20`. Highlight 1 is represented by `&1C`. Highlight 2 is represented by `&1D`.

Printer driver format

The use of a printer driver to support special functions goes hand in hand with the `HT` command. When a highlight character that has been inserted into the text is encountered on printing, it is converted to a special print code. This code may be set for each of the highlight characters using the `HT` command. It is then up to the individual printer drivers to interpret this.

The default codes are:

Highlight 1	128	underline
Highlight 2	129	bold

The code may not be set to less than 128. User-defined special functions must begin at 150. The special codes below 150 will be allocated to different printer functions. 128 and 129 are already allocated and standard printer drivers for printers which support these functions should use them. For the two functions above, highlight characters would come in pairs – one to turn underline on and one to turn it off. This is not mandatory for other, different functions.

Printer driver assembly language format

Printer drivers reside at location `&400`. They may be up to 256 bytes long. Each driver must begin with a jump table.

```
0    JMP    character output
3    JMP    turn printer on
6    JMP    turn printer off
9    JMP    set horizontal motion index
&C   JMP    return option byte
```

Character output

This routine must save all registers, output the character in the `A` register and support any highlight functions which it can.

Turn printer on

This routine must set up the Machine Operating System as required for the given printer and initialise the driver. All registers may be destroyed.

Turn printer off

This routine must turn the printer off and enable output to VDU if it was previously turned off. This may be called when the printer is already switched off. All registers may be destroyed.

Set horizontal motion index (microspacing)

If the printer does not support microspacing, this jump must direct to an `RTS`. Otherwise it should set the HMI of the printer. The HMI is passed in `X`, `X` being the binary number of the 1/120th inch increments for all characters. `X` and `Y` must be preserved.

Return options

This call returns options provided in the driver. It should set the bit(s) in `Y` corresponding to the options available:

Bit	*Option*
0	Microspacing

If no options are available, Y may be returned unaltered (JMP to an RTS). X should be preserved.

All routines are called with JSR and must end with an RTS.

Extended highlights

If you need only underlined type, bold type and microspacing you will not need to reset the default highlights 128 and 129 at all. However, if you have specified highlights such as superscripting and subscripting, italics and extended characters in your printer driver you will need to reset HIGHLIGHT 2. The method is to use the stored command HT with the syntax

```
HT 2 13Ø
```

This switches the HIGHLIGHT 2 code sent by VIEW to the printer driver from 129 to 13Ø. The effects available will be as follows:

Highlight	*Effect*
-	Switch underlining on or off. Automatically cancelled at end of line.
***	Switch bold on or off.
*-	Begin subscripting. Automatically cancelled at end of line.
**	Begin superscripting. Automatically cancelled at end of line.
*	Next character is part of the extended character set.
**-	Return to normal printing after subscripting or superscripting.
*--	Switch alternative font on or off.
-	Switch italics on or off.
*---	Reset all functions to off. You should use this at the start of all documents.
--	This is used by ViewIndex only.

Colours

If you use a colour monitor or a television set you may wish to take advantage of alternative screen colours.

Both text and background colours can be changed in all screen modes except mode 7. The colours available are:

Ø – black 4 – blue

1 – red	5 – magenta
2 – green	6 – cyan
3 – yellow	7 – white

These may be used as follows, in the command screen.

For green text.
Hold down **CTRL** and press `S`.
Press `7 2 0 0 0`

For white text on a blue background.
Hold down **CTRL** and press `S`.
Press `0 4 0 0 0`

For black text on a white background.
Hold down **CTRL** and press `S`.
Press `0 7 0 0 0`
Hold down **CTRL** and press `S`.
Press `7 0 0 0 0`

The first digit indicates whether it is text or background colour that is to be changed. `0` means background and `7` means text. The second digit indicates the colour, as in the table above.

Error messages

The following error messages are given by VIEW.

Message	Description
`Bad file`	The file is too big to be loaded as a printer driver. Printer driver files are up to 256 bytes long.
`Bad filename`	The filename is longer than 19 characters.
`Bad flag`	You have made an error when typing in the flags for the `SETUP` command. The only valid flags are `F`, `J` and `I`.
`Bad marker`	You have referred to a marker which does not exist. The only markers are numbers 1 to 6.
`Bad mode`	You have not specified a positive number for the screen mode.
`Driver does not support microspacing`	The printer driver you are using does not support microspacing so you cannot use the `MICROSPACE` command.

`File not found`	The file you have specified is not in the current directory
`Marker not set`	You have not set a marker before using a command which refers to a marker.
`Mistake`	VIEW does not understand what you have typed.
`Nested macro call`	You are trying to call a macro from inside another macro. VIEW will not accept this.
`Not all read in`	You have used the `READ` command and there is not enough memory for all the text. VIEW reads in as much as it can.
`Not enough memory`	There is not enough memory to load a file. If you do not have shadow memory, you can get more memory by changing mode. If you are working in mode 3, for example, change to mode 7.
`No string found`	VIEW has not found the target string specified in the `SEARCH`, `REPLACE` or `CHANGE` commands.
`No target given`	You have not given a target string with `SEARCH`, `CHANGE` or `REPLACE`.
`No text`	There is no text in memory. You have loaded a non-VIEW file.

Index

Note: names of keys and commands are in capitals.