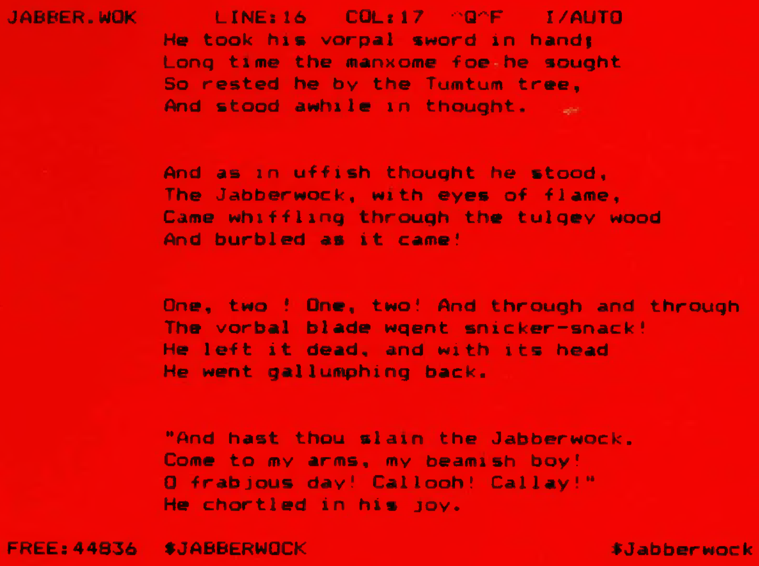
ED80



```
JABBER.WOK       LINE:16   COL:17   ^Q^F   I/AUTO
             He took his vorpal sword in hand;
             Long time the manxome foe he sought
             So rested he by the Tumtum tree,
             And stood awhile in thought.


             And as in uffish thought he stood,
             The Jabberwock, with eyes of flame,
             Came whiffling through the tulgey wood
             And burbled as it came!


             One, two ! One, two! And through and through
             The vorbal blade wqent snicker-snack!
             He left it dead, and with its head
             He went gallumphing back.


             "And hast thou slain the Jabberwock.
             Come to my arms, my beamish boy!
             O frabjous day! Callooh! Callay!"
             He chortled in his joy.

FREE:44836   $JABBERWOCK                       $Jabberwock
```

ED80 is a full screen text editor designed to help you create and amend source programs and other texts quickly and efficiently.

Please take time to read carefully through this manual; there are sections on installing ED80 on your particular computer, the commands available from ED80, some technical details of ED80 and a tutorial; this tutorial is well worth working through even if you have used screen editors before.

The software described by this manual will run on computers with the following specification:

- *a Z80 microprocessor*
- *the CP/M 2.2 disc operating system*
- *a TPA of at least 16K for the ED80 programs*
- *a TPA of at least 36K for the installation programs*

Please check that your system conforms with the above requirements and that you have been supplied with a disc that is correctly formatted for your disc system; the format supplied is shown on the disc label together with your registration number.

Also note that your computer must be running under the CP/M 2.2 operating system before any of the ED80 programs will work.

We strongly encourage you to make one back-up copy of ED80 before using it.

If you cannot get ED80 to run correctly on your computer then please contact your supplier for help or contact HiSoft directly:

HiSoft
180 High Street North
Dunstable LU6 1AT
(0582) 696421

CONTENTS

SECTION 1 ABOUT ED80

ED80 is a full-screen program editor. It is a tool to enable you to type programs into your machine, to inspect and edit them at will and to save them on disc, and has been designed to make the process of program development as easy and convenient as possible. When you use **ED80**, you might immediately notice that it behaves in approximately the same way as a word-processor. Although there are important differences between a program-editor and a word-processor (reflected in the design of **ED80**) the initial set-up of the commands in **ED80** is compatible with the most familiar and widely-used of word-processors, Wordstar. **ED80** has been written, however, so that you may easily tailor the program to your personal preference (See **INSTALLING ED80**) and thus where the manual might read:–

Now move the cursor up one line ([CTRL]–X)

a space has been left after the Wordstar command ([CTRL]–X) so that if you decide to change it using the **ED80INST** program you can fill in the new command.

The supplied disc contains at least six (and possibly seven) files concerned with **ED80**:–

1) ED80.COM	**(The program editor itself)**
2) ED80.HLP	**(The ED80 help file created by ED80INST)**
3) ED80INST.COM	**(The INSTALL program)**
4) ED80INST.MSG	**(The message file for the INSTALL program)**
5) JABBER.WOK	**(Example text file)**
6) EXTRA.WOK	**(Example text file)**
[7) something.E80	**(An optional installation file)]**

It is recommended that the first-time user work through section 2 of the manual (which is not intended to be a work of reference, but rather, a tutorial guide), after which he/she will be familiar enough with **ED80** to use the program without further reference to documentation.

SECTION 2 USING ED80

2.1 GETTING STARTED

You enter **ED80** simply by typing ED80. You may, however, add a filename preceded by a space eg.. ED80 MYFILE.TXT. This will cause that filename to become the Current File. If the Current File exists then it is loaded from the disc otherwise the user simply starts from scratch with an empty file.

Type:–

```
ED80 JABBER.WOK [ENTER]
```

(from now on the symbol [ENTER] will be used to represent a key that may be labeled **ENTER** or **RETURN** or **CR** on your machine). This will activate **ED80** and load the file **JABBER.WOK** from the disc.

2.2 THE STATUS-LINES

You should now be in **ED80** with some text filling the screen. If there appear to be problems at this stage then **ED80** may require installing on your terminal (See **INSTALLING ED80**). The status-lines are described here for an 80 column screen, but even if your console is different (eg.. 40 columns), the differences in the status-lines will not be very significant.

The upper status-line (the top line of the screen) has six sections and should appear like this:–

```
JABBER.WOK    LINE:1    COL:1       INSERT
```

On the left is the name of the Current File. To the right is the line and column at which the cursor is positioned. The space to the right of the column number is for the commands you give to **ED80** and is blank because you haven't issued any yet. To the right of this is the mode (see next section) and finally there is a large space for **ED80**'s messages.

The lower status-line (the bottom line of the screen) has three sections and should appear like this:–

```
FREE:XXXXX   $                    $
```

On the left is the amount of free space you have left in memory which will vary depending on the machine and the size of the current text. The value is approximately equal to the number of characters you can type before **ED80** becomes full of text. The two $ signs mark the start of the Find and Substitute strings which are currently undefined.

On a 40 column screen, the space for the commands is on the top status line at the far left and the messages will overwrite the line and column information.

2.3 THE TEXT WINDOW

The screen in **ED80** is best looked upon as a window onto the current text. This window may be moved in all four directions so that you can view any part of the file. If the window moves downwards then the text appears to move upwards and this is called upward scrolling. In the same way, if the window moves to the right then the text appears to move to the left (leftward scrolling). In **ED80** the scrolling is handled automatically and intelligently to give you the most convenient window onto the text.

2.4 TYPING MODES

There are two basic typing modes: **INSERT** and **CHANGE. INSERT** mode is the more normal method of text entry. When you type a character in **INSERT** mode, all of the line to the right of the cursor is moved right one position before the character is entered. This means that the current line becomes longer by one character. In **CHANGE** mode, the character you type overwrites the current character (ie.. the one at the cursor) and thus the line remains the same length. **ED80** will not allow you to go over the end of the line in **CHANGE** mode.

In general, **INSERT** mode is used to build up a file and **CHANGE** mode is used to alter small sections within a line.

2.5 A TUTORIAL IN THE USE OF ED80

If you are quite familiar with the use of word-processors then a quick glance over this tutorial to clarify the points of difference with the one you are used to should be sufficient. If, however, you are in any doubt as to the full capabilities of a word-processor then it is strongly recommended that you work through this tutorial on your own terminal.

On the screen at the moment is the text of **JABBERWOCKY** from Lewis Carroll's Alice Through the Looking-glass. This is just a sample piece of text to enable you to become familiar with **ED80**. As a re-assuring start, verify that the help key ([CTRL]–J) will list the various commands. Remember that [CTRL]–J means control–J ie.. hold down the control key and then press J. After a quick look at the help page, pressing [ENTER] will get you back to the file.

An important factor in the efficient use of full-screen editors is gaining confidence in moving the cursor around the file. It is worthwhile experimenting with the two main cursor-moving commands which are:- character left ([CTRL]-S) and character right ([CTRL]-D). If you keep pressing the character-right key until the cursor is just past the Y of JABBERWOCKY in the title, and then press it a further time, you'll notice that the cursor "wraps around" to the start of the next line. Also notice how the line and column numbers change in the upper status-line. If character spaces were the only way you could move the cursor then it would take a very long time to get to the end of some files and so, of course, there are many other ways to position the cursor. You can move the cursor one word to the left ([CTRL]-A) or one to the right ([CTRL]-F) or even straight to the beginning ([CTRL]-Q S) or end ([CTRL]-Q D) of the line. Next move the cursor down ([CTRL]-X) and place it at line 7 column 1 just below All mimsy . . . etc. . . . As you may have realised, there is a line missing from the poem and the fourth line of the first stanza should read:

And the mome raths outgrabe.

Type this line in preceded by the correct number of spaces and press [ENTER] at the end. You may have noticed that the second line of this verse is also incorrect and should read:

Did gyre and gimble in the wabe;

Move the cursor up to where the word the has been missed out ie.. the space after in and simply type the word the. Note that the rest of the line is moved to the right after each letter. This is because **ED80** is in **INSERT** mode (note the word on the upper status-line). You can change (or toggle) the typing mode by pressing [CTRL]-V. Now the word **CHANGE** appears in the upper status-line. Move the cursor to the letter o of wobe which should be an a and simply type the letter a. Note that in **CHANGE** mode, the text is not moved to the right and the characters typed simply overwrite the existing ones. If you finish off the line by typing be; then you can also see that **ED80** will not allow you to overwrite [ENTER] in **CHANGE** mode.

A word of explanation is in order here about [ENTER]. It is a character as much as is the letter "A" and can be cursored-over, deleted, found and substituted (as [CTRL]-M) as can any other character. In fact the installation program gives you the option of displaying all [ENTER]s so you can see where you are more easily. The most usual way of displaying them is <.

The second verse has been omitted from the poem and this is a good excuse to test the deleting commands in **ED80** and the auto indent feature. Firstly go into **INSERT** mode ([CTRL]-V) and then move the cursor to the start of the blank line below at the end of the first verse. Now press [ENTER] twice to give a good separation between the stanzas. You can toggle auto-indent by pressing [CTRL]–O I ie.. [CTRL]–O and then I (or i or [CTRL]–I or [CTRL]–i) and the message I/AUTO will appear on the upper status-line. Now type the first line preceded by the correct number of spaces so that the line starts directly under those of the first verse:

Beware the Jabberwock, my son!

If you make a mistake then you can delete the character before the cursor ([DEL]) or the character at the cursor ([CTRL]–G). Press [ENTER] at the end of the line and note that the new line starts immediately under the one above. This is due to the auto-indent and is an extremely useful feature when writing programs to improve legibility. Now type in the other three lines of the stanza ending all with [ENTER]:

The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!

Typing mistakes are usually very common when entering programs and **ED80** has been designed to minimise the effects of the more common errors. Thus whenever you delete a line in **ED80** it is stored **UNTIL YOU START TO EDIT ANOTHER LINE** and can be recovered. To illustrate this, move the cursor until it rests on the line starting Beware the Jabberwock... and give the command to delete the line ([CTRL]–Y). As you can see it disappears from the text, but the restore line command ([CTRL]–O R) can be used to get the line back and in fact you may move the cursor to an entirely different place and "restore" the line again as many times as required. This manoeuvre can be quite useful for moving a line or copying it to some other place in the text. If you start editing a line ie.. type in a few characters to an existing line and then use the restore line command, the line is restored to what it was originally.

In a large program using the Find and Substitute facilities in **ED80** is often the best way of getting to a known point in a program. To illustrate this, position the cursor on the first line of the poem and give the find first command ([CTRL]–Q F). This puts the cursor just after the first dollar sign on the lower status-line. Now type in Jabberwock. (the word Jabberwock followed by a full-stop) (you can use the delete character left ([DEL]) if you make a mistake) and then [ENTER] and just press [ENTER] for the second or Substitute string. The cursor will now be positioned on line 34. Note that the two previous occurrences of Jabberwock are not followed by a full-stop. If you now put the cursor to the top of the file ([CTRL]–Q R) and redefine the Find string as Jabberwock then the cursor will first rest on line 10. Now the find next command ([CTRL]–L) will go to line 23.

The next part of the tutorial illustrates the way you can manipulate blocks of text, rather than just characters and lines. To define a block of text you have to mark its start and end. The original Lewis Carroll poem duplicates the first verse at the end and so the aim is to mark the whole of the first verse as a block and then copy it to the end. However, this will be done in rather an unusual way to illustrate the block buffer in **ED80**. Put the cursor to the start of line 4 ie.. the start of the first line of the first verse and mark this point as the start of a block ([CTRL]–K B). Now position the cursor at the end of the last line of the poem (the space after outgrabe.) and mark this as the end of the block ([CTRL]–K K). Now the unusual part: delete this block ([CTRL]–K Y)!! You can now see that there is a star after the figure showing the amount of free space left on the lower status-line. This star means that there is a block in the block buffer. You can see the size of the block by operating the free-space toggle ([CTRL]–O F). You get the block back by using the paste block command ([CTRL]–O P). Do this now. Note that the block is still there and can be pasted as many times as required. Now move the cursor to the end of the file ([CTRL]–Q C) and paste the block again and lo! the last verse is duplicated. Exactly the same effect would have been achieved by marking the block and then copying it ([CTRL]–K C) except that the block would not have been put in the buffer. Both block delete and block move (which is exactly equivalent to delete followed by paste) put the block in the buffer if there is space.

It is possible to write a block to the disc ([CTRL]–K W) and read a block from the disc. As an exercise, move the cursor right to the end of the file and then issue the command to read a block ([CTRL]–K R). You are now prompted for the filename of the file you wish to read in. Type:–

```
EXTRA.WOK [ENTER]
```

(you can use [DEL] if you make a mistake typing in the name). Naturally Alice did not understand the explicit and obscure sexual connotations of the poem as we do today and the poem stands as an interesting if rather distressing insight into Dodgson's dark, tulgey mind.

Finally, it is very important to be able to save an edited file to the disc. There are three ways to quit **ED80**. Firstly, one may abandon the file ([CTRL]–K Q). Here nothing is saved and the current text is lost. Then one may want to save the file with no backup ([CTRL]–O Q). This will save the current text on the disc deleting a file of the same name if it existed. This method is generally used when space is low on the disc. Finally, the most normal method of quitting **ED80** is to save with a backup ([CTRL]–K X). Here, if there is a file with the same name it is converted to type **.BAK** and thus is preserved. Issue the save with backup command ([CTRL]–K X). You are prompted for a filename, with the Current Filename already given for convenience. The Current Filename may be deleted if you require and the text saved under another name. In this case, however, just press [ENTER]. When silence and the A> prompt rules again, a look at the disc directory with DIR will show that the original file is now called JABBER.BAK and there is a new JABBER.WOK that is the file we have just edited.

It should now be perfectly possible (with frequent forays into the help-pages) to edit your own files. Before doing so it is advisable to cast a quick glance over the reference section as there are some very useful features documented there that have not been covered in this tutorial.

SECTION 3 INSTALLING ED80

The process of installing **ED80** involves three phases. **ED80** is first read in from the disc. Then, sections of the program are modified and finally **ED80** is written back out to the disc (as a **.COM** file) together with a help (**.HLP**) file. Thus the process involves a permanent change to **ED80**.

There are two reasons that you might want to install **ED80**. Primarily, it may be that there are problems with the screen layout and **ED80** seems not to work at all. This will be due to incorrect terminal codes and in this case you should read the section on **TERMINAL INSTALLATION**. Alternatively, you may wish to modify some of the commands or options to suit either keyboard or taste. This procedure is covered in the section **RE-DEFINING ED80 COMMANDS** and **RE-DEFINING USER OPTIONS**. In either case you should first read the next section.

3.1 STARTING UP THE INSTALL PROGRAM

To run the installing program, insert the supplied disc and type:–

```
ED80INST [ENTER]
```

You will now see the **ED80INST** copyright message and some general information. When you're ready, press any key. The purpose of the installation process is to alter the copy of **ED80** on the disc. To this end, some copy of the program (called the working copy) is read in from the disc into the machine. The first question is thus:–

```
Normally the working copy of ED80 is
read in from a file called ED80.COM
Use another file instead (Y/N)   ?
```

The reply will normally be N, the exception being when you have renamed a version of **ED80**. A reply of Y will produce the prompt:–

```
[ESC] to abort
Omit file type (.COM assumed)
Enter filename
```

to which a filename should be typed in (omitting the filetype eg . . . to use **EDIT.COM** as the image, type EDIT [ENTER]). If you typed Y by mistake and really do want to use ED80.COM as the working copy then just type ED80. Whether you replied N to the opening question or Y and then specified a filename, the working copy will now be read in to the machine from the disc and the **ED80** Installation Menu will appear.

There is now a copy of **ED80** in the memory of your machine ready to be altered and the **ED80** Installation Menu on the screen.

```
ED80 INSTALLATION MENU
--------------------
1. Return to CP/M
2. Alter screen codes
3. Save ED80 as <working copy filename> (normally
   ED80.COM)
4. Save ED80 as another file
5. Alter command codes
6. Alter user options
7. Load installation from .E80 file
8. Save installation to .E80 file

Type desired number:
```

If you are a first-timer using the installation program because the screen codes in **ED80** were wrong then turn first to the section **TERMINAL INSTALLATION** and then to **LEAVING THE INSTALL PROGRAM**. The other sections in this chapter are **USER PATCHES**, **REDEFINING ED80 COMMANDS**, **REDEFINING USER OPTIONS**, and **USE OF INSTALLATION FILES.**

3.2 TERMINAL INSTALLATION

Select option 2 from the main menu to alter the screen codes. You will be asked

```
How many screen columns (     ) ?            and then
How many screen rows (     ) ?
```

in answer to each of these questions you should type in the correct number followed by [ENTER]. Pressing [ENTER] alone is equivalent to giving the answer in brackets.

The rest of the questions concern how the screen controller works on your machine. If you are in doubt about any of the questions, consult the manual for your machine. You are now asked for the:–

```
Cursor position lead-in sequence
(     ) (     )-
```

When **ED80** is in operation it has to be able to tell the screen controller to put the cursor at a certain position on the screen. To do this, **ED80** tells the controller the row and the column required. Most screen controllers require a special sequence of codes to indicate that the values to follow represent a row and a column. Thus inside the first set of brackets there will be the sequence as it is currently defined with the decimal values of the codes in that sequence in the second set of brackets. If the sequence is correctly set up then just press [ENTER] and move on to the next question. If the sequence is incorrect then it must be changed and there are two ways to change it.

1) If your screen controller does not have a special sequence of codes then you will have to write a program in assembly language to do it instead. In this extremely unlikely event, press D to delete the current sequence.

2) If, as is much more likely, your screen controller does have a Cursor Position lead-in sequence then you should enter it now code by code (up to a maximum of four codes) terminated by [ENTER]. Each code may either be entered as a single keypress or as its decimal value terminated by [ENTER]. As an example, if the correct sequence for your controller was [CTRL]–K =. You could enter this either by typing

[CTRL]–K = [ENTER] (ie.. 4 keypresses) or by typing
1 1 [ENTER] 6 1 [ENTER] [ENTER]
([CTRL]–K is ASCII 11 and = is ASCII 61 and note the two [ENTER]s at the end. The first is to terminate the 61 and the second is to terminate the whole sequence.)

The next question asked is

```
Is the row sent before the column (      )
                            (Y/N/[ENTER])  ?
```

The screen controller may require the row before the column, or the column before the row. As above, pressing [ENTER] is equivalent to giving the answer in brackets.

You are now asked

```
Offset for column (     )  ?          and then
Offset for row (     )  ?
```

When the values for the row and the column are sent, many screen controllers require an offset to be added to each. The values required for the offsets are those required to position the cursor at the top left of the screen (ie.. if the correct offsets for your machine were both 32 then sending the Cursor Position lead-in sequence, then 32. then 32 will put the cursor at the top left of your screen). If the value in brackets is correct then just press [ENTER] otherwise type in the correct value terminated by [ENTER]. As above, you should consult the manual for your machine if in any doubt.

The next text to appear is:–

```
Clear Screen sequence
(       )   (       )-
```

The layout is identical with that for the cursor positioning sequence detailed above. Press [ENTER] alone if the sequence for clearing the screen is correct or enter the correct sequence terminated by [ENTER] as above. If your controller does not recognize a sequence to clear the screen (possible but unlikely) then press D.

```
Clear to End of Line Sequence
(       )   (       )-
```

prompts you for the sequence to clear to the end of the current line. Respond to the prompt exactly as above for the clear– screen sequence. It is quite possible that your screen controller does not recognize a sequence for clearing to the end of the current line. If this is so then press D to delete the sequence and **ED80** will perform the function itself by software (although more slowly than the controller would do it).

```
Use lead-in (      )
Use lead-out (      )
          (Y/N/[ENTER])  ?
```

The final questions concern the use of lead-in and lead-out sequences. These options allow you to use **ED80** to send a command to the screen controller or run a small program at the start and end of an editing session. For example, this facility might be used to put your machine into 80 column mode for editing and reset back to 40 column mode on exit from **ED80**. However, unless you have an important reason for wanting to use this facility, it is advisable to answer N to both questions. If you answer Y to either you will be asked to specify a code sequence to send to the screen controller which you should enter as described above. If however, you wish to do something more complicated than just send a sequence then you should press D to delete the current sequence and prepare to write a program for the patch file!

Normally you will now be returned to the main menu. If, however, you do not see the main menu now then read on!

3.3 USER PATCHES

A user-patch is a program written by you in assembly language to perform a function not within the capabilities of your screen controller. They will be needed if your response to certain questions from option 2 of the main menu has been to press D ie.. the screen controller cannot perform certain functions. The functions which may need user patches are:– Cursor Position, Clear Screen, Lead-in, and Lead-out. If you have answered D to any of these then after the last question of option 2, you will see the message

```
Please read the manual (Section 3) ! (which you are. Good)
User patch area starts at #XXXX

Read in a new Patch file
                    (Y/N)  ?
```

The normal process by which you can write a user-patch is to use an assembler (**GEN80** I hope) to write and assemble the program and create a .COM file, which is the form needed for the patch. Included in this manual is an assembly language source file that is extensively commented to illustrate the general format for a patch-file. If you need to write a patch-file then reply N to the question for the moment and study the example patch-file closely. If you have already written and assembled the patch-file then reply Y to the question above and then in response to the prompt type in the filename. The machine-code in the patch-file will then be incorporated from the disc into the memory copy of **ED80** and you will then be returned to the main menu. It is possible to get the message

```
Too many characters in commands
```

when trying to read in the patch-file. This message means that the size of the command definitions and the patch-file combined is too large. You should try to change the command definitions to be shorter and thus quicker (both to type and to execute) and more efficient.

3.4 RE-DEFINING ED80 COMMANDS

Pressing 5 from the main menu will allow you to alter the command definitions. All of the commands will be shown and you have the opportunity to change the definition or accept it and pass on to the next command. After the last command you are returned to the main menu. For each of the commands the display format is:–

```
Command name
(keystroke definition) (decimal definition) –
```

where the keystroke definition is the sequence of keys the user presses to give the command and the decimal definition is the decimal **ASCII** value of those keys.

At any stage you have the option to go back to consider the previous command, to retain the current definition or to change the current definition.

1) To backtrack to the previous command, press B

2) To retain the current definition press [ENTER]. The process then repeats for the next command. At the end you are returned to the main menu.

3) To change the current definition the new definition should be typed in element by element (up to a maximum of four elements) and terminated by [ENTER] after which the redefined command appears. If you are now sure that the definition is correct then press [ENTER] to pass to the next command, otherwise type in another definition and the whole process is repeated.

4) Definition elements are of two types. The first type is simply a keystroke and the second type is a sequence of digits terminated by [ENTER]. For example, the two ways to include a [CTRL]-Y (which has an ASCII value of 25) in the definition are:-
a) hold down the [CTRL] key and press Y
b) press 2 then 5 then [ENTER]
The two modes of entry of elements may not be mixed within the same definition. Thus if the first character of a definition was a number then all subsequent numbers are treated as their numerical value. However, if the first character was not a number then all subsequent numbers are treated as **ASCII** characters. This feature is included so that command definitions such as [CTRL]-K 0 can be entered as three key presses (ie.. hold down [CTRL] and press K, then press 0)

If the definition given is the same as that of a previous command or a prefix to a previous command then this message will appear:-

```
WARNING : There is a conflict between the
--------------------
and --------------------commands.
Do you wish to continue anyway (Y/N)  ?
```

A response of Y will ignore the duplication and N will allow the current command to be re-defined. Note that if **ED80** is saved to the disc with two commands identical, the use of one of the commands will be lost.

It is recommended that you consult the reference section of the manual if in any doubt as to the meaning of some of the commands. After the last command, you are returned to the main menu.

3.5 RE-DEFINING USER OPTIONS

You can change the user options by selecting 6 from the main menu.

There are four user options. They are:–

```
Size of tabs (      ) ?
```

to which you should type in the tab size required followed by [ENTER] or [ENTER] alone to retain the value in brackets.

```
Tabs per scroll (      ) ?
```

When the cursor in **ED80** moves off the right-hand edge of the screen, the text window moves to the right (ie.. the text appears to scroll to the left). This left/right scroll works in units of one tab. On most screens, a value of one or two is best for this parameter. Enter the value as above.

```
End of line display
(      )   (      ) ?

End of file display
(      )   (      ) ?
```

A single key response is needed for both of these. If you don't wish the end of lines or the end of file to be displayed then press D to delete the current value, otherwise type the character you would like to be used for each (< is a common end-of-line marker with maybe | for end-of-file). Although not normally used in word-processors, the markers can be useful in a program editor for distinguishing spaces and tabs at the end of lines and end of file. After responding to these options you are returned to the main menu.

3.6 USE OF INSTALLATION FILES

There are many features of **ED80** that are alterable by the user. Every copy of **ED80** naturally contains one set of these options. There is a type of file, however, called an Installation File that consists solely of the set of the alterable options. An Installation File is of type **.E80**.

To save the current installation information in a file, select option 7 from the main menu. You will then be prompted for a filename which you should type in terminated by [ENTER]. It is possible that you will see the error message

```
Too many characters in commands
```

If so, you should decrease the number of characters used to define the commands or the size of the patch-file (if you're using one).

To load an installation file, select option 8 from the main menu. As above, you will be prompted for a filename. If the file you give does not exist then the prompt will be repeated. You can press [ESC] to quit. When the installation file is loaded into memory, it will overwrite the alterable options already present in the copy of **ED80** in memory.

The main use of Installation Files is when you are in the long-term process of tailoring your version of **ED80** to suit your own preferences. If you save each successive change you make to the installation of **ED80** then any changes you find undesirable can be overwritten by using the last installation file rather than going all the way through the commands. You may also find it useful to save your final installation in a file as a reminder of how your commands are defined.

3.7 LEAVING THE INSTALL PROGRAM

You can leave the install program by selecting option 1 from the main menu, but **BEWARE!** If you select option 1 then nothing will be changed on the disc. Thus if you are satisfied with the changes you have made in the last installation session, you should first use either option 3 or option 4. Both will save a copy of **ED80** (as a **.COM** file) and a help file (as a **.HLP** file) on the disc. Option 3 will save both files under the name you specified at the beginning of the session (normally **ED80**) whereas option 4 allows you to change the name by which you will invoke **ED80**. You may have more than one copy of **ED80** on the disc at the same time (under different names, of course) without a clash of help files.

Thus the normal method of leaving the install program will be first to select option 3 and then option 1. If you don't wish to save the results of your installing labours then select option 1 alone.

N.B. You may, when saving the copy of **ED80** get the error message

```
Too many characters in commands
```

in which case you should either decrease the size of your command definitions, or the size of the patch-file or both. It is well worth spending some time deciding on the design of the command definitions. A well-designed and succinct set will be easier to use and will also lead to quicker and more efficient editing.

SECTION 4 COMMAND REFERENCE GUIDE

This Section is intended as a short reference guide to the commands and features of **ED80**. In all cases, the default command is given in brackets followed by a space so that the user may fill in any new commands. Where possible, the default command is the same as the Wordstar command. A ? means that there is no exact Wordstar equivalent for this command and the sign [CTRL]– means that the control key is held down. [ENTER] indicates that the user should press the requisite key, whereas <CR> indicates a byte of the value £OD (ASCII 13).

4.1 CURSOR-MOVING COMMANDS

Character left/right ([CTRL]–S : [CTRL]–D)
Move the cursor one character position left/right. Moving past the end of a line positions the cursor at the beginning of the next line. Likewise moving past the beginning of a line puts the cursor at the end of the previous line. (This feature is hereafter called wraparound).

Character left (alt) ([CTRL]–H=backspace)
As in Wordstar there are two cursor left commands.

Word left/right ([CTRL]–A : [CTRL]–F)
Move the cursor to the beginning of the last/next word. Characters that constitute the boundaries between words are:– "()[]{}=+– \ */<> ↑–;:,£$ [TAB] and wraparound operates.

Tab left/right ?([CTRL]–O S : [CTRL]–O D)
Move the cursor to the last/next tab position. The size of tab stops may be defined by the user (see **INSTALLING ED80**). Wraparound operates.

Start/End of line ([CTRL]–Q S : [CTRL]–Q D)
Move the cursor to the start/end of the current line. Wraparound does not operate.

Line up/down ([CTRL]–E : [CTRL]–X)
Move the cursor up/down one line. After moving up or down one line, the cursor column is always the same. Thus it may appear that the cursor is positioned beyond the end of a line. If another line up/down or page up/down command is issued then the cursor will move as described. However, if any other key is pressed, **ED80** will behave as though the cursor was at the end of the current line (Ambiguous cursor).

Top/Bottom of screen ?([CTRL]–O E : [CTRL]–O X)
Move the cursor to the top/bottom of the screen.

Page up/down ([CTRL]–R : [CTRL]–C)
Move the text window down/up by one less than the number of non-status lines displayed on the screen. Thus a page up command on a 25 line screen will move the text window up by 22 lines (25 screen lines-2 status lines-1) and the old top line becomes the new bottom line. Ambiguous cursor operates

Start/End of file ([CTRL]–Q R : [CTRL]–Q C)
Move the cursor to the start/end of the file.

4.2 TEXT DELETING COMMANDS

Delete line ([CTRL]–Y)
Delete the current line. Note that the line is placed into the editing buffer and can be recalled into the text by use of the restore line command. The deleted line will be overwritten when the user next makes a change to any line.

Delete last character ([DEL])
Delete the character to the left of the cursor. Wraparound operates.

Delete this character ([CTRL]–G)
Delete the character under the cursor. Wraparound operates.

Delete word left/right (?[CTRL]–O T : [CTRL]–T)
Delete from the cursor to the beginning of the last/next word. The characters that constitute the boundaries between are given under the Word left/right command above. Wraparound operates.

Delete to start of line ([CTRL]–Q [DEL])
Delete from the cursor to the beginning of the current line.

Delete to end of line ([CTRL]–Q Y)
Delete from the cursor to the end of the current line.

4.3 BLOCK COMMANDS

Mark start/end of block ([CTRL]–K B : [CTRL]–K K)
Place the block markers. A marker will be positioned at the cursor position. The markers are lost if the line containing the marker is altered subsequently.

Move block ([CTRL]–K V)
Delete the currently marked block from the text and place in the block buffer, then insert the block at the cursor position. If there is enough space the block will be retained in the buffer, but the less space there is, the longer the command will take.

Copy block ([CTRL]–K C)
Copy the currently marked block from the text to the cursor position.

Delete block ([CTRL]–K Y)
Delete the currently marked block from the text and place in the block buffer. The less space there is, the longer this command will take. This is due to the procedure required to place the block in the buffer rather than abandoning it altogether. Thus, if the amount of free space is very small (less than 256) the user is asked whether to abandon the block. If the user presses Y then the block will be deleted from the text and not placed in the block buffer. If the user does not want to completely abandon the block then N should be pressed, the block should be written out to the disc (from where it may later be read back in if desired) and then deleted.

Paste block ?([CTRL]–O P)
Insert at the cursor the block currently in the block buffer. The block remains in the buffer if there is sufficient space.

Read block ([CTRL]–K R)
The user is asked for a filename. [ENTER] alone aborts the command. A filename followed by [ENTER] will search the disc for the filename given and insert it at the cursor. The response RDR: will read the block from the current logical reader deviee.

Write block ([CTRL]–K W)
The user is asked for a filename. [ENTER] alone aborts the command. A filename followed by [ENTER] will write the currently marked block to the disc with the filename given.

Printing a Block
In response to the filename prompt, LST: will send the block to the current logical list device and may thus be used to print a block of text. The response PUN: will send the block to the current logical punch device. The whole file may thus be printed by setting the block markers to the start and end of the file and writing the block to LST: (but see **Printing the File** below).

4.4 QUICK CURSOR MOVEMENT

Goto line ?([CTRL]–O G)
User will be prompted for a line number. This should be entered digit by digit (the **DELETE CHAR LEFT** command may be used as a destruc-

tive backspace) and after [ENTER] the cursor will be positioned at the start of the line given. This command is extremely convenient for quick access to an error reported by a compiler or assembler.

Goto start/end of block ([CTRL]–Q B : [CTRL]–Q K)
Move the cursor to the start/end block marker.

Remember position ([CTRL]–K O)
The current cursor position is stored. The marker is lost if the line in which it lies is subsequently changed.

Return to position ([CTRL]–Q O)
The cursor is positioned at the stored position.

4.5 FIND AND SUBSTITUTE

Find first ([CTRL]–Q F)
The current Find string is displayed. [ENTER] will retain the current string, otherwise the user should type in the required Find string (up to a maximum of 32 characters) and then press [ENTER]. The **DELETE CHAR LEFT** command may be used as a destructive backspace, [CTRL]–R will redisplay the previous string and [CTRL]–U will abort the operation leaving the strings as they were. A control character may be entered by pressing the control meta-key ([CTRL]–P) (see **MISCELLANEOUS**) and then the control character (eg.. [CTRL]–P then [ENTER] enters a <CR> or [CTRL]–M into the string). Pressing the meta-key and then ? will return a value which is displayed as ? and counts as a wild-character when in the Find string.

After [ENTER] is pressed the operation is repeated for the Substitute string, and then the cursor is positioned at the start of the first occurrence of the Find string in the file.

Find next ([CTRL]–L)
The file is searched for the next occurrence of the Find string starting from one character after the cursor. A wild-character in the Find string will match with any character at all in the file.

Substitute and find ?([CTRL]–O L)
The file is searched for the next occurrance of the Find string starting from the cursor. A wild-character in the Find string will match with any character at all in the file. When the string is found, it is replaced by the Substitute string and the cursor is positioned after the last character of the Substitute string. Finally the file is searched for the next occurrence of the Find string starting from the cursor.

Substitute all ?([CTRL]–O A)
Starting from the cursor, all occurrances of the Find string in the file are replaced by the Substitute string. A wild-character in the Find string

will match with any character at all in the file. The cursor is then placed after the last string substituted.

4.6 LEAVING ED80

Quit and Exit ([CTRL]–K Q)
The user is asked whether to abandon the file. Pressing Y will cause a return to CP/M and the current text will be abandoned. Pressing [SPACE] will prompt the user for a filename and the given file will be loaded from the disc thus effectively also abandoning the current text. Any other response will abort the command.

Edit without backup ?([CTRL]–O Q)
The Current Filename is displayed after the prompt Filename:. This may be deleted using the **DELETE CHAR LEFT** command ([DEL]) and altered, or the whole command can be aborted by pressing [CTRL]–U. When the user is satisfied with the filename, [ENTER] or [SPACE] will cause the current text to be saved on the disc under the filename given. If [ENTER] is used as the terminator then after saving the file the user is returned to **CP/M**. After [SPACE], however, the user is prompted for a filename and then will be able to edit another file without leaving **ED80**. Thus the normal response after this command will be [ENTER] or [SPACE] alone which will save the file with its original name. A file already on the disc with the same name will be lost.

Exit with a backup ([CTRL]–K X)
Identical with above except that a file already on the disc with the same name as the Current Filename will be renamed as a .BAK file and any .BAK file with the same name will be lost.

Printing the File
In response to the filename prompt, LST: will cause the current text to be written to the current logical list device and may thus be used to send a file to the printer. PUN: will write the text to the current logical punch device. Note that after both these responses the user will abandon the current text and the disc copy of the Current Filename will be unaltered. A better way of printing the whole file is to set the block markers at the start and end of the file and then write the block to LST: (see **Printing a Block** above).

4.7 TOGGLES

Toggle mode ([CTRL]–V)
Switch between **INSERT** and **CHANGE** mode. A character typed in **INSERT** mode will only be entered after the characters to the right of the cursor on the same line have been moved right one character position. A character typed in **CHANGE** mode will overwrite the current character. A <CR> may not be overwritten in **CHANGE** mode.

Toggle auto indent ?([CTRL]–O I)
Auto indent will only operate in **INSERT** mode. The message INSERT will become I/AUTO. When indent is on and [ENTER] is pressed in **INSERT** mode, the next line will be indented so that it starts at the same column as the line above.

Toggle free space ?([CTRL]–O F)
A star following the amount of free space indicates that there is a block in the block buffer. The free space toggle is used to check the size of the block.

4.8 MISCELLANEOUS

Deliver tab ([CTRL]–I)
Will return a tab character ([CTRL]-I or ASCII 9). The size of tabs and thus the position of tab stops may be defined by the user. Tabs will be entered as a ASCII 9 in the file and will not be changed to spaces. They are treated in the main like any other character in the file.

Restore line ?([CTRL]–O R)
If the user is in the process of editing a line then this command will restore the line to what it was when the user first positioned the cursor on it. If the user is not in the process of editing a line then this command will insert in front of the current line, the last edited line. This aspect of the command is useful because the **DELETE LINE** command places the deleted line into the line-buffer exactly as though it had just been edited. The user may thus move a line from one place to another by deleting it, moving the cursor to the desired place and then issuing a **RESTORE LINE** command.

Disc directory ?([CTRL]–K F)
The prompt Filename: is given. See **RULES FOR FILENAMES.** A reply of [ENTER] or [SPACE] alone will abort the command as will pressing [CTRL]–U. After the filename is terminated by [ENTER] or [SPACE], the screen is cleared and a directory is printed (in fact the directory given will be the same as that seen after the equivalent DIR command). Any key will then return the user to the current text.

Erase file ([CTRL]–K J)
The prompt Filename: is given. See **RULES FOR FILENAMES.** A reply of [ENTER] or [SPACE] alone will abort the command, as will pressing [CTRL]–U. After the filename is terminated by [ENTER] or [SPACE] the named file or files will be deleted from the disc.

Control meta-key ([CTRL]–P)
Any key pressed after the meta-key will be entered into the file as its literal value. This may thus be used to enter control characters into the file that are normally commands or prefixes to commands (eg..

[CTRL]–P then backspace enters a [CTRL]–H). The meta-key can also be used in the same way to enter control characters in the Find and Substitute strings. In this case if ? is pressed after the meta-key a character is returned that is displayed as ? but acts as a wild-character in the find string.

Help key ?([CTRL]–J)

Pressing the help key will display help pages giving information on the commands available from **ED80** and how to access them.

SECTION 5 PROMPTS AND MESSAGES IN ED80

There are four prompts produced by **ED80**. They appear on the lower status line. Two require a single key response and the other two require a string of characters terminated by [ENTER].

Abandon block: Sure?

This prompt requires a single character response. It appears if the user has issued the **DELETE-BLOCK** command and there are less than 256 bytes free. If the user responds Y then the block will be deleted and lost (note that the block is normally saved in the block buffer and thus not lost) while any other response will abort the command.

The prompt also appears if the user has issued any command the execution of which would overwrite the block in the block buffer. If the user responds Y then the block in the block buffer will be lost and the command executed while any other response will abort the command.

Abandon text: Sure?

This prompt requires a single character response. It is produced after the **QUIT AND EXIT** command. If the user responds Y then the current text will be lost and the user returned to CP/M. If the user responds [SPACE] then he will be further prompted to produce a filename (see later) and the named file will be loaded and the old current text abandoned. Any other response will abort the command.

Filename:

This prompt requires a string of characters terminated by [ENTER] or [SPACE]. It is produced after any command that requires reading from or writing to the disc. The response is interpreted as a filename and the maximum allowed length is 14 characters (See **RULES FOR FILENAMES**). In building up the filename, the currently defined **DELETE CHARACTER LEFT** can be used as a destructive backspace. When the two **EXIT** commands are issued, the Current Filename is given for convenience although it may be deleted if desired and another name substituted. If the name returned is null ie.. [ENTER] or [SPACE] alone, or [CTRL]–U is pressed at any stage then the command is aborted. There are three responses to this prompt that are not interpreted as a filename, viz LST: PUN: RDR: These address the logical list device, the logical punch device and the logical reader device respectively.

Go to line:

This prompt requires a string of numbers terminated by [ENTER]. It is produced after the **GO TO LINE** command. The response is interpreted as a line number and the maximum allowed length is 4 characters (only numbers are accepted). In building up the number the currently defined **DELETE CHARACTER LEFT** can be used as a destructive backspace. [ENTER] alone aborts the command.

RULES FOR FILENAMES

A filename consists of three fields. The drivename, the filename and the filetype. (eg.. B: MYFILE .GEN).

When giving a filename:–

1) The drivename is optional and if not given the current logged-in drive is assumed.

2) In a command where ambiguous filenames are allowed (ie.. **ERASE FILE** and **DISC DIRECTORY**) a ? may be used to represent any single character and a * may be used as if the remainder of the field in which it occurs (barring the drivename field) were filled out with ?s.

3) In the directory command a response of the drivename field alone is interpreted as though the filename and filetype were *.

For example:–

B:MYFILE.* Addresses files of any filetype on disc B called MYFILE

*.BAK Addresses all files of type .BAK on the current drive

FILE*.GEN Addresses files of type .GEN on the current disc whose filename starts with the letters FILE

FILE?.GEN Addresses files of type .GEN on the current disc whose filename contains 5 letters and starts with the letters FILE

B: or B:*.* The first form (drivename alone) can only be used for the **DISC DIRECTORY** command. Addresses all files on drive B

There are thirteen messages produced by **ED80** and they appear on the lower status-line as do the prompts.

Out of memory

Indicates that there is not enough space in the machine to carry out the proposed command.

Line is too long

Produced when the length of the line would exceed the maximum allowed length (255 characters) if the proposed action was taken. This might either be simply the press of a key, or the deletion of a <CR>.

Undefined command

Indicates that the initial key of a command is correct, but the second or subsequent keys do not form a valid command.

Block start unmarked/Block end unmarked

Produced after any block operation if the start/end of the block has not been marked or the mark has been lost (ie., the line containing the mark has been edited).

Block marks reversed

Produced after any block operation if the start of the block occurs after the end.

Invalid destination

Produced after a **MOVE BLOCK** or **COPY BLOCK** and indicates that the destination (cursor) lies between the start and end of the block.

Block too big

Produced after a **READ BLOCK** command and indicates that the file on the disc is too large to fit into memory.

No block in buffer

Produced after a **PASTE BLOCK** operation and is self-explanatory.

Marker lost

Produced after a **RETURN TO POSITION** command and indicates that the position marker has not been placed or has been lost.

No file/Bad filename

Produced after any command which prompts the user for a filename. The command indicates either that the filename is badly formed or inappropriate or the file does not exist.

Disc full

Produced after any command that tries to write to the disc. Indicates that either the disc or the disc directory is full. The user should consider using the **ERASE FILE** command.

No such line

Produced after the **GO TO LINE** command and indicates that the line number given is greater than the number of lines in the file.

SECTION 6 TECHNICAL DETAILS

6.1 INTERNAL FILE FORMAT

Text in **ED80** is held simply as a string of ASCII characters. The end-of-line sequence is <CR> (ASCII 13) rather than <CR> <LF>, allowing the user greater text space. The end-of-file is marked by a <NULL> (ASCII 0). When the text is written to the disc, however, <CR> is replaced with <CR><LF> and the <NULL> is replaced by [CTRL]-Z (ASCII 26) thus making the disc file written by **ED80** compatible with normal **CP/M** text files.

The maximum line-length in **ED80** is 255 characters. Note that the cursor column number may exceed 255 due to tab and control characters (in which case the column number displayed on the status-line remains 255).

The maximum number of lines in the file is limited only by memory considerations, but note that the line number display on the status-line is only of four digits (due to space considerations) and the **GO TO LINE** command can only reach line 9999 and no further. All other commands will work as normal if the number of lines exceeds 9999 (although note also that on most systems the average number of characters per line would have to be about three for the line numbers to exceed 9999).

6.2 NON-PRINTING CHARACTERS

Characters of ASCII value less than 32 decimal (Control characters) are treated as far as the user is concerned as any other character. They may be entered into the file by first pressing the control meta-key ([CTRL]-P) and then the control character desired. If the terminal is capable of producing characters of value greater than 127 decimal, then these characters are entered as any other into the file and are displayed as ?

The meta-key may also be used to specify control characters in the find and substitute strings in the same way as above. An obvious use for this feature is to find the end-of-line character (reached by [meta-key] [ENTER] or [meta-key][CTRL]-M). The ? character when pressed after the meta-key returns £80. This character is displayed as ? in the find and substitute strings, but is treated as a wild-character in the find string ie.. it will match with any character at all in the file. (Note that if a terminal has a key that can return £80 then this will be in all respects identical to [meta-key] followed by ?).

6.3 DATA AND EXTERNAL DEVICES

Whenever the user gives a command that would normally access the disc (ie.. **READ BLOCK, WRITE BLOCK, EXIT WITHOUT BACKUP, EXIT WITH A BACKUP**) there are three responses to the prompt Filename: that are interpreted as logical external devices.

1) LST: if used for a write operation will send the data to the current logical list device, which is normally a printer (but may, of course, be set from **CP/M** using STAT). When the data is sent a <LF> character (ASCII 10) is sent after every <CR> as is usual for **CP/M** files.

2) PUN: if used for a write operation will send the data to the current logical punch device. As above, every <CR> is sent as <CR><LF>, but unlike the use of LST: a [CTRL]-Z (ASCII 26) is sent after the data to mark the end-of-file. [CTRL]-Z is the standard **CP/M** end-of-file character.

3) RDR: when used for a read operation is designed to be compatible with PUN: or indeed any standard **CP/M** data transfer operation. The top bit of every character is reset (thus masking out any parity bits sent by the transmitting hardware) and all <LF> characters are ignored to produce the standard **ED80** internal format. RDR: requires a [CTRL]-Z character to mark the end-of-data. Files may thus easily be transferred from one machine to another from inside **ED80** by use of PUN: and RDR:.

SECTION 7 EXAMPLE PATCH-FILE

```
;This is an example patch-file for use with ED80 or MON80.
;The patch-file for MON80 must be position independent, whereas that
;for ED80 need not be.  This example is written to be position independent
;so that it can be used for both programs.

;=========================================================================
;If this is a patch-file for ED80 (which does not have to be position-
;independent) and the file includes some position-dependent code eg.. a
;CALL to within itself or a LD  instruction addressing an area within
;itself then there should be an ORG statement here at the head of the file.
;The value for the ORG is that given by the installing program in the line
;         User Patch Area starts at #XXXX
;so assuming the #XXXX was #2438 the statement should be
;         ORG   #2438
;Note that the value given is adjusted for the initial length byte
;=========================================================================


LENGTH     DEFB FINISH-START
;The first byte of the file must be the total length
;of the patch-file excluding itself (255 maximum).


START
          JR    CLEAR_SCREEN
;Jump relative to the routine to clear the screen

          JR    CURSOR_POSITION
;Jump relative to the routine to position the cursor

          DEFS  2
;The vector to your lead-in routine would go here, but in this example
;none is required.  The two bytes should be filled however.

          DEFS  2
;A lead-out routine is not used in this example.  See above.

;The file should start with four two-byte areas that are either vectors
;to the routines concerned or uninitialised space (DEFS 2).  The way ED80
;and MON80 uses the vectors is simply to CALL the requisite location
;ie.. where the vector to the routine is positioned.  If the user has
;specified in the installing program that a certain patch is not required
;then the vector will never be called.  Common sense may be used; as an
;example, if the only patch required was to clear the screen, then the patch
;file need only contain the length byte followed by the routine itself as
;the vectors for CURSOR_POSITION, LEAD_IN and LEAD_OUT are never called.
```

```
CLEAR_SCREEN
;This routine must clear the whole screen, but need not adjust the
;cursor position, or do anything else normally associated with a
;screen-clearing routine.  It takes no parameters and may corrupt all
;registers except IX.  Use of IY and the alternate register set is not
;advisable as the BIOS of some machines corrupt these.

          CALL #F060      ;This is an example
          LD   HL,#FFC8   ;of a possible screen
          SET  3,(HL)     ;clearing routine.  Note that
          RET             ;it should finish with a simple RET
CURSOR_POSITION
;This routine must set the cursor position on the screen.
;The routine takes two parameters, the column and the row thus:-

;B holds the desired row    + offset for row
;C holds the desired column + offset for column

;The routine may corrupt all registers except IX.  Use of IY and the
;alternate set is not advisable for the reason given above.

          LD   A,B
          LD   (#FFD0),A ;An example routine
          LD   A,C        ;to position the cursor
          LD   (#FFD2),A ;Note that the
          PUSH IX         ;routine saves the IX
          CALL #F076      ;register which is
          POP  IX         ;destroyed by the CALL
          RET             ;and then does a simple RET


;Lead-in and lead-out are not required in this example.  Code need
;not, of course, be included.  The only requirement is that there be
;two bytes where the vector ought to be.   See above

;NB the code for the lead-in and lead-out may corrupt all registers
;and should end with a simple RET

FINISH    EQU  $
```

The questions we are asked most often

Here are the answers to the questions we get asked most frequently - if you get stuck read this:

Q. Why do I get error 3 (undeclared identifier) when I use one of SIN, COS, TAN, ARCTAN, EXP, LN, RANDOM, RANSEED or FRAC ?
A. You haven't used the command line option T. See Page P-3.

Q. Why do I get BDOS ERROR ON A: R/O ?
A. You're using CP/M 2 and have changed discs without typing [CTRL] C at the A> prompt.

Q. My program looks O.K. but the compiler doesn't generate a .COM file and finishes with the error 'No more text'.
A. The last line of your program needs to finish with 'END.' and a newline. Like many CP/M programs (e.g. Digitial Research's SUBMIT, RMAC) Pascal80 must have a CR/LF before the end of file. You should edit your program; go to the end of text (with [CTRL] Q [CTRL] C) and type ENTER.

Q. When I load ED80 I get rubbish on the screen as well as the text. How do I make it look alright?
A. ED80 isn't installed for your screen. Run the ED80 install program as described in the ED80 manual Page E-9 and change the screen codes. If you don't know the screen codes, look in the documentation that came with your computer. If this doesn't help contact your dealer or the computer's manufacturer.

Q. I seem to have installed ED80 OK but if the cursor is on the bottom line and I do cursor down, then the display gets messed up.
A. It sounds as if you have the number of lines on the screen installed incorrectly; check this carefully.