```
C1    No                                                          F17
TxtL                                    <R>                       65254
Next: Default Format to be Carriage return                        Auto.
      A      B      C          D          E       F
  1 Init|Name    |<No >Road          |Town      |$  |
  2     |        |   |               |          |   |
  3 R.T.|Cowan   |16 |Jeremy Grove   |Hampton   | 17|
  4 G.L.|Blake   |37 |Osmaston Road  |Knowle    | 56|
  5 A.J.|Hunt    |25 |Latimer Road   |Wythall   | 43|
  6 H.A.|Fisher  |32 |Florence Road  |Henley    | 28|
  7 C.D.|Beard   |86 |Valley Road    |Redditch  | 25|
  8     |        |   |               |          |   |
  9     |        |   |               |          |   |
 10     |        |   |               |          |   |
 11     |        |   |               |          |   |
 12     |        |   |               |          |   |
 13     |        |   |               |          |   |
 14     |        |   |               |          |   |
 15     |        |   |               |          |   |
 16     |        |   |               |          |   |
 17     |        |   |               |          |   |
```

Now two further such columns are needed, one in front of the town column and one at the end in front of the $ column:

```
H1                                                                I17
Car    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                   65254
Next:                                                             Auto.
      A      B    C  D          E          F       G     H  I
  1 Init|Name    | |No |Road          | |Town      |  < >$  |
  2     |        | |   |              | |          |  |  |   |
  3 R.T.|Cowan   | |16 |Jeremy Grove  | |Hampton   |  |  17|
  4 G.L.|Blake   | |37 |Osmaston Road | |Knowle    |  |  56|
  5 A.J.|Hunt    | |25 |Latimer Road  | |Wythall   |  |  43|
  6 H.A.|Fisher  | |32 |Florence Road | |Henley    |  |  28|
  7 C.D.|Beard   | |86 |Valley Road   | |Redditch  |  |  25|
  8     |        | |   |              | |          |  |   |
  9     |        | |   |              | |          |  |   |
```

You only want to print a portion of the sheet and so you use the Copy block option. Note that the Mail labels option intercepts the carriage returns that you have set and works out the way the printout should be organised to produce the desired effect - you don't actually get a new

line started wherever a carriage return has been inserted.

```
H1                                                                I17
Car                crd...crd <R>                                  65254
Next: Copy block A3...H7                                          Auto.
      A      B    C  D          E          F       G     H  I
  1 Init|Name    | |No |Road          | |Town      |  < >$  |
  2     |        | |   |              | |          |  |  |   |
  3 R.T.|Cowan   | |16 |Jeremy Grove  | |Hampton   |  |  17|
  4 G.L.|Blake   | |37 |Osmaston Road | |Knowle    |  |  56|
  5 A.J.|Hunt    | |25 |Latimer Road  | |Wythall   |  |  43|
  6 H.A.|Fisher  | |32 |Florence Road | |Henley    |  |  28|
  7 C.D.|Beard   | |86 |Valley Road   | |Redditch  |  |  25|
  8     |        | |   |              | |          |  |   |
  9     |        | |   |              | |          |  |   |
```

Note that it is ABSOLUTELY ESSENTIAL for you to make sure that the end of the range is on a Carriage Return Format column as with cell H7 above. Omitting this requirement will do no harm but you will not get the correct print formatting. By typing M2 (for Mail labels, 2 across) after the request for destination you will be set up for having mail labels printed two abreast:

```
H1                                                                I17
Car                                        0-9, <R>               65254
Next: Destination Mail Labels, groups of 2                        Auto.
      A      B    C  D          E          F       G     H  I
  1 Init|Name    | |No |Road          | |Town      |  < >$  |
  2     |        | |   |              | |          |  |  |   |
  3 R.T.|Cowan   | |16 |Jeremy Grove  | |Hampton   |  |  17|
  4 G.L.|Blake   | |37 |Osmaston Road | |Knowle    |  |  56|
  5 A.J.|Hunt    | |25 |Latimer Road  | |Wythall   |  |  43|
  6 H.A.|Fisher  | |32 |Florence Road | |Henley    |  |  28|
  7 C.D.|Beard   | |86 |Valley Road   | |Redditch  |  |  25|
  8     |        | |   |              | |          |  |   |
  9     |        | |   |              | |          |  |   |
```

Here is what they will look like. Not very organised, but your next task is to change the width of the columns so that various parts line up under one another:

G.L. Blake A.J. Hunt
37 Osmaston Road 25 Latimer Road
Knowle Wythall

H.A. Fisher C.D. Beard
32 Florence Road 86 Valley Road
Henley Redditch

R.T. Cowan
16 Jeremy Grove
Hampton

After some width adjustment, here is what you can achieve. In this case the labels are going to be printed three abreast:

```
G1    Town                                                          I17
TxtL              crd...crd <R>                                     65254
Next: Copy block A3...H7                                            Auto.
      A          B        C  D          E          F          G      H  I
   1 Init|Name           | |No |Road          | <Town              > |$  |
   2     |               | |   |              | |                  | |   |
   3 R.T.|Cowan          | |16 |Jeremy Grove  | |Hampton           | | 17|
   4 G.L.|Blake          | |37 |Osmaston Road | |Knowle            | | 56|
   5 A.J.|Hunt           | |25 |Latimer Road  | |Wythall           | | 43|
   6 H.A.|Fisher         | |32 |Florence Road | |Henley            | | 28|
   7 C.D.|Beard          | |86 |Valley Road   | |Redditch          | | 25|
   8     |               | |   |              | |                  | |   |
   9     |               | |   |              | |                  | |   |
```



```
G1    Town                                                          I17
TxtL                                        0-9, <R>                65254
Next: Destination Mail Labels, groups of 3                          Auto.
      A          B        C  D          E          F          G      H  I
   1 Init|Name           | |No |Road          | <Town              > |$  |
   2     |               | |   |              | |                  | |   |
   3 R.T.|Cowan          | |16 |Jeremy Grove  | |Hampton           | | 17|
   4 G.L.|Blake          | |37 |Osmaston Road | |Knowle            | | 56|
   5 A.J.|Hunt           | |25 |Latimer Road  | |Wythall           | | 43|
   6 H.A.|Fisher         | |32 |Florence Road | |Henley            | | 28|
   7 C.D.|Beard          | |86 |Valley Road   | |Redditch          | | 25|
   8     |               | |   |              | |                  | |   |
   9     |               | |   |              | |                  | |   |
  10     |               | |   |              | |                  | |   |
  11     |               | |   |              | |                  | |   |
  12     |               | |   |              | |                  | |   |
  13     |               | |   |              | |                  | |   |
  14     |               | |   |              | |                  | |   |
  15     |               | |   |              | |                  | |   |
  16     |               | |   |              | |                  | |   |
  17     |               | |   |              | |                  | |   |
```

G.L. Blake A.J. Hunt H.A. Fisher
37 Osmaston Road 25 Latimer Road 32 Florence Road
Knowle Wythall Henley

C.D. Beard R.T. Cowan
86 Valley Road 16 Jeremy Grove
Redditch Hampton

Further adjustment may be needed with extra lines at the end, or changes in the spacing to suit your particular labels - this is acheived most simply by trial and error.

Note that if you had used the simple Print option rather than Mail labels the Carriage Return columns will have produced unpredictable and probably undesirable results.

Recap: We have seen how the Copy command is also used for saving and loading data on the disk drive and also for printing the data.

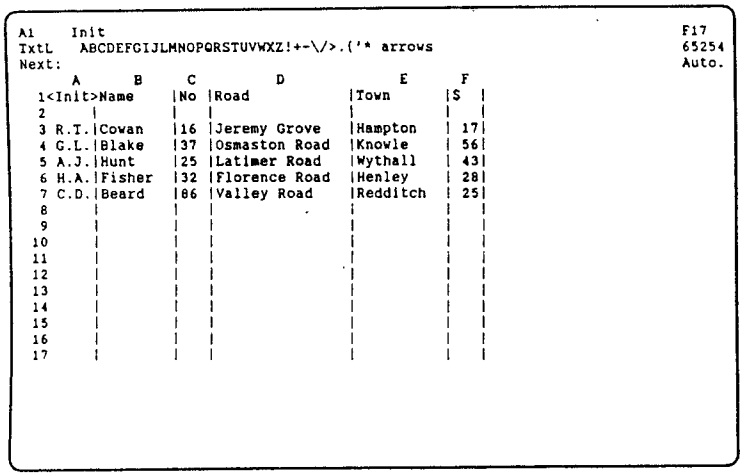
When saving or loading the choice of filename extension signals to THE CRACKER which type of data it should send or expect.

When printing you may have to make use of the OUT command and a special Carriage Return format in order to get the desired effect.

PARTITIONING THE SCREEN

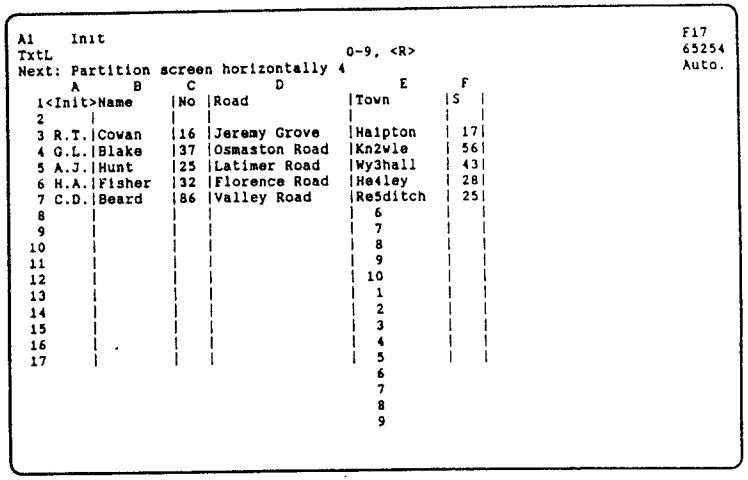
As your worksheet gets larger you will, at times, want to work somewhere like the bottom right hand corner but still see your line and column headings which are at the top or left of the sheet. You can do this with the partition commands. These allow you to split the screen either horizontally, vertically or both so that you get two, or four smaller displays. The usual term for such split screens is 'windows'.

As well as moving the windows independently you also have the option of locking them so that as you move the cursor in one part, the other will move in a synchronized way. For example, if you scroll the lower window the appropriate headings will scroll past in the top of the screen. Cursor control can be moved between the windows with single key commands.

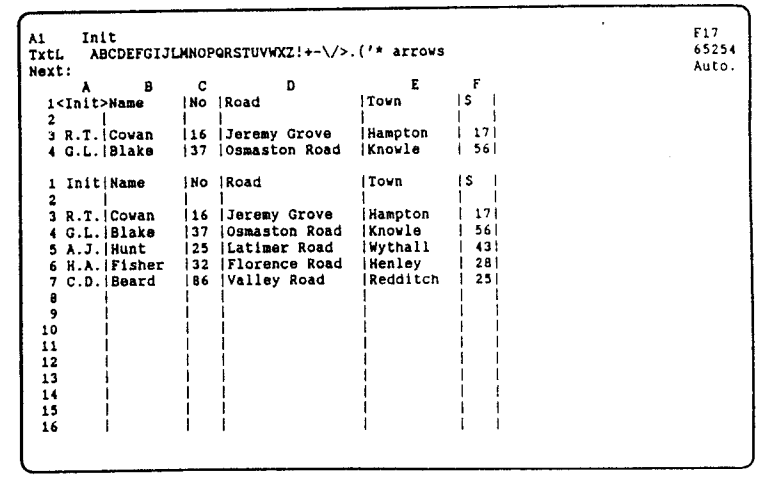


	A	B	C	D	E	F
1	<Init>	Name	No	Road	Town	$
2						
3	R.T.	Cowan	16	Jeremy Grove	Hampton	17
4	G.L.	Blake	37	Osmaston Road	Knowle	56
5	A.J.	Hunt	25	Latimer Road	Wythall	43
6	H.A.	Fisher	32	Florence Road	Henley	28
7	C.D.	Beard	86	Valley Road	Redditch	25
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

First you are going to see the screen partitioned horizontally. As you type PH, for Partition Horizontally, you will see that a grid is put up on the screen. You must use this to judge where you want the split to take place. The number on the grid that you choose will be the first location of the second window on the screen.



	A	B	C	D	E	F
1	<Init>	Name	No	Road	Town	$
2						
3	R.T.	Cowan	16	Jeremy Grove	Hailpton	17
4	G.L.	Blake	37	Osmaston Road	Kn2wle	56
5	A.J.	Hunt	25	Latimer Road	Wy3hall	43
6	H.A.	Fisher	32	Florence Road	He4ley	28
7	C.D.	Beard	86	Valley Road	Re5ditch	25
8					6	
9					7	
10					8	
11					9	
12					10	
13					1	
14					2	
15					3	
16					4	
17					5	
					6	
					7	
					8	
					9	



	A	B	C	D	E	F
1	<Init>	Name	No	Road	Town	$
2						
3	R.T.	Cowan	16	Jeremy Grove	Hampton	17
4	G.L.	Blake	37	Osmaston Road	Knowle	56

1	Init	Name	No	Road	Town	$
2						
3	R.T.	Cowan	16	Jeremy Grove	Hampton	17
4	G.L.	Blake	37	Osmaston Road	Knowle	56
5	A.J.	Hunt	25	Latimer Road	Wythall	43
6	H.A.	Fisher	32	Florence Road	Henley	28
7	C.D.	Beard	86	Valley Road	Redditch	25
8						
9						
10						
11						
12						
13						
14						
15						
16						

You should now have two sets of row numbers starting from 1. Don't be confused by seeing the same information in both parts of the screen, it merely means that both windows are set to the same part of the worksheet. You can leave the top view where it is and on a big worksheet move the bottom view far down the sheet.

The next command to investigate is the backslash \. This is used to jump between horizontal window sections.

```
A1    Init                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A      B     C          D              E       F
  1<Init>Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|

  1 Init|Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5 A.J.|Hunt     |25 |Latimer Road    |Wythall    | 43|
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8      |        |   |                |           |   |
  9      |        |   |                |           |   |
 10      |        |   |                |           |   |
 11      |        |   |                |           |   |
 12      |        |   |                |           |   |
 13      |        |   |                |           |   |
 14      |        |   |                |           |   |
 15      |        |   |                |           |   |
 16      |        |   |                |           |   |
```

```
A1    Init                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A      B     C          D              E       F
  1 Init|Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|

  1<Init>Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5 A.J.|Hunt     |25 |Latimer Road    |Wythall    | 43|
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8      |        |   |                |           |   |
  9      |        |   |                |           |   |
 10      |        |   |                |           |   |
 11      |        |   |                |           |   |
 12      |        |   |                |           |   |
 13      |        |   |                |           |   |
 14      |        |   |                |           |   |
 15      |        |   |                |           |   |
 16      |        |   |                |           |   |
```


The effect as you can see is to jump to the same cell but in the other window. Now let us see what happens when we move the cursor in the lower part and then transfer back to the upper part.

```
A5    A.J.                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A      B     C          D              E       F
  1 Init|Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|

  1 Init|Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5<A.J.>Hunt     |25 |Latimer Road    |Wythall    | 43|
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8      |        |   |                |           |   |
  9      |        |   |                |           |   |
 10      |        |   |                |           |   |
 11      |        |   |                |           |   |
 12      |        |   |                |           |   |
 13      |        |   |                |           |   |
 14      |        |   |                |           |   |
 15      |        |   |                |           |   |
 16      |        |   |                |           |   |
```

Now the transfer back using the same keystroke.

```
A1    Init                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A      B     C          D              E       F
  1<Init>Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|

  1 Init|Name     |No |Road            |Town       |$  |
  2      |        |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5 A.J.|Hunt     |25 |Latimer Road    |Wythall    | 43|
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8      |        |   |                |           |   |
  9      |        |   |                |           |   |
 10      |        |   |                |           |   |
 11      |        |   |                |           |   |
 12      |        |   |                |           |   |
 13      |        |   |                |           |   |
 14      |        |   |                |           |   |
 15      |        |   |                |           |   |
 16      |        |   |                |           |   |
```

The cursor has not moved in the upper screen because initially the screen parts were not locked together. To lock them use the PSH<R> (Partition Synchronized Horizontal) command sequence:

```
A1    Init                                                          F17
TxtL                          Vert, Horiz                           65254
Next: Partition synchronised                                        Auto.
      A       B      C          D              E        F
   1<Init>Name     |No |Road             |Town       |$   |
   2      |        |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|

   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|
   5 A.J.|Hunt     |25 |Latimer Road     |Wythall    | 43|
   6 H.A.|Fisher   |32 |Florence Road    |Henley     | 28|
   7 C.D.|Beard    |86 |Valley Road      |Redditch   | 25|
   8     |         |   |                 |           |    |
   9     |         |   |                 |           |    |
  10     |         |   |                 |           |    |
  11     |         |   |                 |           |    |
  12     |         |   |                 |           |    |
  13     |         |   |                 |           |    |
  14     |         |   |                 |           |    |
  15     |         |   |                 |           |    |
  16     |         |   |                 |           |    |
```

To test it you can move the cursor down and then swap parts:

```
A3    R.T.                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A       B      C          D              E        F
   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3<R.T.>Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|

   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|
   5 A.J.|Hunt     |25 |Latimer Road     |Wythall    | 43|
   6 H.A.|Fisher   |32 |Florence Road    |Henley     | 28|
   7 C.D.|Beard    |86 |Valley Road      |Redditch   | 25|
   8     |         |   |                 |           |    |
   9     |         |   |                 |           |    |
  10     |         |   |                 |           |    |
  11     |         |   |                 |           |    |
  12     |         |   |                 |           |    |
  13     |         |   |                 |           |    |
  14     |         |   |                 |           |    |
  15     |         |   |                 |           |    |
  16     |         |   |                 |           |    |
```

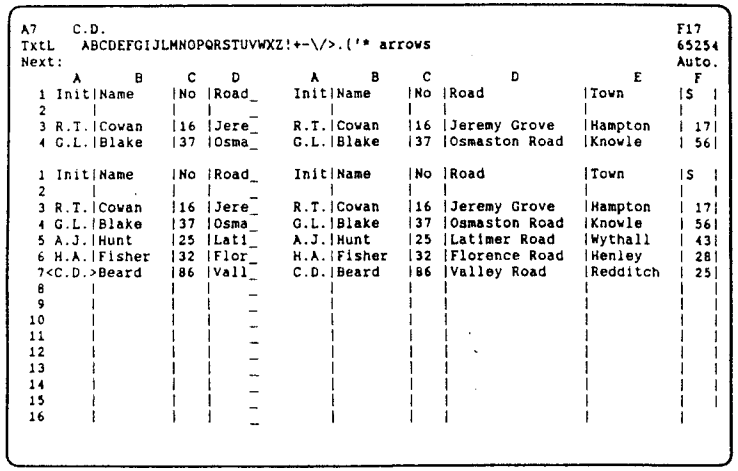

Now the swap:

```
A7    C.D.                                                          F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65254
Next:                                                               Auto.
      A       B      C          D              E        F
   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|

   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|
   5 A.J.|Hunt     |25 |Latimer Road     |Wythall    | 43|
   6 H.A.|Fisher   |32 |Florence Road    |Henley     | 28|
   7<C.D.>Beard    |86 |Valley Road      |Redditch   | 25|
   8     |         |   |                 |           |    |
   9     |         |   |                 |           |    |
  10     |         |   |                 |           |    |
  11     |         |   |                 |           |    |
  12     |         |   |                 |           |    |
  13     |         |   |                 |           |    |
  14     |         |   |                 |           |    |
  15     |         |   |                 |           |    |
  16     |         |   |                 |           |    |
```

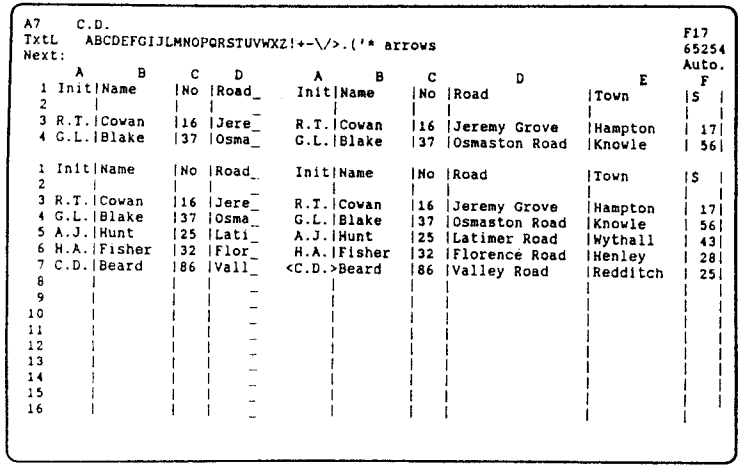
The lower cursor has also moved down two places. The screen can also be divided vertically in a very similar way using PV (Partition screen Vertically) and specifying the grid number.

```
A7    C.D.                                                          F17
TxtL                              0-9, <R>                          65254
Next: Partition screen vertically 20                                Auto.
      A       B      C          D              E        F
   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston Road    |Knowle     | 56|

   1 Init|Name     |No |Road             |Town       |$   |
   2     |         |   |                 |           |    |
   3 R.T.|Cowan    |16 |Jeremy Grove     |Hampton    | 17|
   4 G.L.|Blake    |37 |Osmaston 2oad    |Kno3le     | 54|        5         6
   5 A.J.|Hun1234567890123456789012345678901234567890123456789012345678901
   6 H.A.|Fisher   |32 |Florence Road    |Henley     | 28|
   7<C.D.>Beard    |86 |Valley Road      |Redditch   | 25|
   8     |         |   |                 |           |    |
   9     |         |   |                 |           |    |
  10     |         |   |                 |           |    |
  11     |         |   |                 |           |    |
  12     |         |   |                 |           |    |
  13     |         |   |                 |           |    |
  14     |         |   |                 |           |    |
  15     |         |   |                 |           |    |
  16     |         |   |                 |           |    |
```

As before a grid will be superimposed:



There are now four parts to the screen. To move across the vertical parts you use the forward slash /:



You can synchronize the parts vertically or cancel synchronisation at any time. To get rid of the partitioning just type PE for Partition End. If you save the worksheet on file it will be in the unpartitioned form.

It is after using the Partition command that you may find the CTRL-P option to produce a screen copy the most useful.

Recap: After entering headings or explanatory comments into one part of a sheet it is often desirable to be able to still see these even when the cursor has moved to a distant part of the sheet.

The Partition command lets you define up to four separate windows on the sheet which can be locked to scroll together or left independent.

DATABASE HANDLING TECHNIQUES

SEARCHING

You may want to find a particular entry within a large worksheet without having to search for it yourself. To help you there is the GET command which goes through the columns and lines starting at the current cursor location looking for any string that you care to enter. (A string is computer jargon for any row of characters that are to be treated as a whole. Any word in this sentence can be looked on as a string.) Note that it does not matter what format the target string is displayed under, only the data as it was entered is searched.

Your string must be enclosed in single character delineators. Valid delineators that can be used to mark the beginning and end of the string are any characters that are printable but not letters or numerals. The string that you want found will be assumed to have been completed when the second delineator to match the first character after the GET command is found.

If you want to find a second occurrence of the same string then you only need to type a valid delineator twice and the string you last used will be automatically inserted between the characters.

```
A1    Init                                                        F17
TxtL      /string/                                                65254
Next: Get /43                                                     Auto.
      A      B       C           D            E       F
  1 Init|Name     |No |Road            |Town       |S  |
  2     |         |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5 A.J.|Hunt     |25 |Latimer Road    |Wythall    | 43|
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8     |         |   |                |           |   |
  9     |         |   |                |           |   |
 10     |         |   |                |           |   |
 11     |         |   |                |           |   |
 12     |         |   |                |           |   |
 13     |         |   |                |           |   |
 14     |         |   |                |           |   |
 15     |         |   |                |           |   |
 16     |         |   |                |           |   |
 17     |         |   |                |           |   |
```

In this example you are searching for the number 43 which is to be found at location F5.


```
F5    (43)                                                        F17
Int    ABCDEFGIJLMNOPQRSTUVWXZ|+-\/>.('* arrows                   65254
Next:                                                             Auto.
      A      B       C           D            E       F
  1 Init|Name     |No |Road            |Town       |S  |
  2     |         |   |                |           |   |
  3 R.T.|Cowan    |16 |Jeremy Grove    |Hampton    | 17|
  4 G.L.|Blake    |37 |Osmaston Road   |Knowle     | 56|
  5 A.J.|Hunt     |25 |Latimer Road    |Wythall    < 43>
  6 H.A.|Fisher   |32 |Florence Road   |Henley     | 28|
  7 C.D.|Beard    |86 |Valley Road     |Redditch   | 25|
  8     |         |   |                |           |   |
  9     |         |   |                |           |   |
 10     |         |   |                |           |   |
 11     |         |   |                |           |   |
 12     |         |   |                |           |   |
 13     |         |   |                |           |   |
 14     |         |   |                |           |   |
 15     |         |   |                |           |   |
 16     |         |   |                |           |   |
 17     |         |   |                |           |   |
```

The cursor ends up at the location of the string. You should remember that only the actual cell formulae, numeric data and text entries are searched. If for example the last column had been in financial format and you had tried to search for 43.00 you would not find it even though it was displayed. As you can see from the contents line only 43 is actually stored in the memory. In practice this means you can only search for things that can be displayed on the contents line.

The Get command is useful for making long jumps across a complicated sheet. You can insert special text entries as markers which help you to quickly find the right place even if insertions and deletions have been made.

SORTING THE LINES

THE CRACKER can selectively sort lines, with either all or part of a column used as the basis of the sort. Text and numeric data can be sorted in either increasing or decreasing form, allowing easy maintenance of address lists and client lists. By using only part of columns in the sorts you carry out many of the activities that you would otherwise use a database management program for.

```
F5    (43)                                                    F17
Int                              crd...crd <R>                65254
Next: Sort lines using range: B3...B7                         Auto.
      A      B     C         D              E        F
  1 Init|Name    |No |Road            |Town      |$   |
  2     |        |   |                |          |    |
  3 R.T.|Cowan   |16 |Jeremy Grove    |Hampton   |  17|
  4 G.L.|Blake   |37 |Osmaston Road   |Knowle    |  56|
  5 A.J.|Hunt    |25 |Latimer Road    |Wythall   |  43|
  6 H.A.|Fisher  |32 |Florence Road   |Henley    |  28|
  7 C.D.|Beard   |86 |Valley Road     |Redditch  |  25|
  8     |        |   |                |          |    |
  9     |        |   |                |          |    |
```

In this example the lines are going to be sorted using the name as the basis.

```
F5    (43)                                                    F17
Int                              Inc, Dec                     65254
Next: Increasing or decreasing                                Auto.
      A      B     C         D              E        F
  1 Init|Name    |No |Road            |Town      |$   |
  2     |        |   |                |          |    |
  3 R.T.|Cowan   |16 |Jeremy Grove    |Hampton   |  17|
  4 G.L.|Blake   |37 |Osmaston Road   |Knowle    |  56|
  5 A.J.|Hunt    |25 |Latimer Road    |Wythall   |  43|
  6 H.A.|Fisher  |32 |Florence Road   |Henley    |  28|
  7 C.D.|Beard   |86 |Valley Road     |Redditch  |  25|
  8     |        |   |                |          |    |
  9     |        |   |                |          |    |
 10     |        |   |                |          |    |
 11     |        |   |                |          |    |
 12     |        |   |                |          |    |
 13     |        |   |                |          |    |
 14     |        |   |                |          |    |
 15     |        |   |                |          |    |
 16     |        |   |                |          |    |
 17     |        |   |                |          |    |
```



```
B3    Beard                                                   F17
TxtL   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows               65254
Next:                                                         Auto.
      A      B     C         D              E        F
  1 Init|Name    |No |Road            |Town      |$   |
  2     |        |   |                |          |    |
  3 C.D.<Beard   >86 |Valley Road     |Redditch  |  25|
  4 G.L.|Blake   |37 |Osmaston Road   |Knowle    |  56|
  5 R.T.|Cowan   |16 |Jeremy Grove    |Hampton   |  17|
  6 H.A.|Fisher  |32 |Florence Road   |Henley    |  28|
  7 A.J.|Hunt    |25 |Latimer Road    |Wythall   |  43|
  8     |        |   |                |          |    |
  9     |        |   |                |          |    |
 10     |        |   |                |          |    |
 11     |        |   |                |          |    |
 12     |        |   |                |          |    |
 13     |        |   |                |          |    |
 14     |        |   |                |          |    |
 15     |        |   |                |          |    |
 16     |        |   |                |          |    |
 17     |        |   |                |          |    |
```

Column B is now in alphabetical order. Next you will see a numerical sort in descending order. Note that the sort is carried out on the internal value (as displayed on the contents line) of the number and not on the numerals as displayed.

```
B3    Beard                                                   F17
TxtL                             crd...crd <R>                65254
Next: Sort lines using range: F3...F7                         Auto.
      A      B     C         D              E        F
  1 Init|Name    |No |Road            |Town      |$   |
  2     |        |   |                |          |    |
  3 C.D.|Beard   |86 |Valley Road     |Redditch  |  25|
  4 G.L.|Blake   |37 |Osmaston Road   |Knowle    |  56|
  5 R.T.|Cowan   |16 |Jeremy Grove    |Hampton   |  17|
  6 H.A.|Fisher  |32 |Florence Road   |Henley    |  28|
  7 A.J.|Hunt    |25 |Latimer Road    |Wythall   |  43|
  8     |        |   |                |          |    |
  9     |        |   |                |          |    |
```

```
B3    Beard                                                        F17
TxtL                                <R>                          65254
Next: Increasing or decreasing D                                 Auto.
      A      B     C         D              E        F
   1 Init|Name    |No |Road           |Town       |$   |
   2     |        |   |               |           |    |
   3 C.D.|Beard   |86 |Valley Road    |Redditch   |  25|
   4 G.L.|Blake   |37 |Osmaston Road  |Knowle     |  56|
   5 R.T.|Cowan   |16 |Jeremy Grove   |Hampton    |  17|
   6 H.A.|Fisher  |32 |Florence Road  |Henley     |  28|
   7 A.J.|Hunt    |25 |Latimer Road   |Wythall    |  43|
   8     |        |   |               |           |    |
   9     |        |   |               |           |    |
  10     |        |   |               |           |    |
  11     |        |   |               |           |    |
  12     |        |   |               |           |    |
  13     |        |   |               |           |    |
  14     |        |   |               |           |    |
  15     |        |   |               |           |    |
  16     |        |   |               |           |    |
  17     |        |   |               |           |    |
```

```
F3    (56)                                                        F17
Int    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                   65254
Next:                                                            Auto.
      A      B     C         D              E        F
   1 Init|Name    |No |Road           |Town       |$   |
   2     |        |   |               |           |    |
   3 G.L.|Blake   |37 |Osmaston Road  |Knowle     |< 56>
   4 A.J.|Hunt    |25 |Latimer Road   |Wythall    |  43|
   5 H.A.|Fisher  |32 |Florence Road  |Henley     |  28|
   6 C.D.|Beard   |86 |Valley Road    |Redditch   |  25|
   7 R.T.|Cowan   |16 |Jeremy Grove   |Hampton    |  17|
   8     |        |   |               |           |    |
   9     |        |   |               |           |    |
  10     |        |   |               |           |    |
  11     |        |   |               |           |    |
  12     |        |   |               |           |    |
  13     |        |   |               |           |    |
  14     |        |   |               |           |    |
  15     |        |   |               |           |    |
  16     |        |   |               |           |    |
  17     |        |   |               |           |    |
```

The lines have now been sorted to make the numbers in column F descending.

Remember that if you do want to keep address lists you will probably also like to be able to prepare mail labels. See the Copy function for more details on this.
Note that there are one or two important restrictions you should bear in mind when using the Sort command. Consult the complete command reference for more details.

Recap: There are two functions, a search command and a sort command, that allow you to reproduce some simple database handling effects using THE CRACKER.

THE CRACKER TUTORIAL
V. MORE COMPLEX USE OF THE SHEET

MORE ADVANCED EXPRESSIONS AND FUNCTIONS

Here we are going to look at the more advanced Functions available to you for use in your expressions. If you feel that we have already covered all of the inbuilt functions that you can take in during the early stages, then you can skip this part of the manual for now although it is advisable to return to it when you feel more competent.

THE IF,THEN,ELSE FUNCTIONS

This is a special function group which is known in computing terms as a conditional branch. This means that a test is made on some data, and if the data passes (ie the test is TRUE) then one function is carried out, whereas if the data fails (ie the test is FALSE) then a different function is applied.

This conditional function is of the form IF("),THEN("),ELSE("). The " stands for an expression. The first expression, following the IF, must be logical. 'Logical' is another piece of jargon that just means it must have an answer of TRUE or FALSE. An example of a logical expression is IF(B3=4) which has a value of TRUE if B3 does equal 4 or FALSE if it does not.

When the test is TRUE the expression associated with the THEN part is executed and, correspondingly, when the test is FALSE the expression associated with the ELSE part is executed.

The full list of special operators you can use to give you an answer of TRUE or FALSE are:

= equal
: not equal
> greater than
] greater than or equal (certain keyboards only)
< less than
[less than or equal (certain keyboards only)

You can also use the functions TRUE or FALSE themselves instead of an expression. In other words, you can just type the words 'TRUE' or 'FALSE' in as a cell entry, they do not have arguments so nothing else has to be typed into the entry. These logical functions have numeric values of -1 (TRUE) and 0 (FALSE).

Once a cell has been assigned a logical function it can then be referenced as the expression for the IF function of a separate entry. For example, you could set B3 to TRUE or FALSE and then use the conditional in the form IF(B3),THEN("),ELSE("). It should follow from the above explanation that what this means is IF (B3 reads TRUE) THEN ("), ELSE IF (B3 reads FALSE) (").

In place of the normal expressions after THEN and ELSE you can use the special function ERROR. When the Error function is encountered during a calculation then processing is stopped and a message is put up on the prompt line, which can be treated as a normal error message. No harm can be done using this function, but it is a useful method to check for genuine errors, to bring macros to an end, or to warn the user that a certain unwanted result has occurred, for example if profits drop below a certain figure.

You can now try an example which includes some of these functions and features. Clear the worksheet and type IC15<R>G<R> and IL<R><R> to set up your work area. Now type .TRUE<R> into A1:

```
A1                                                              A2
Gen   Enter number or expression                                65519
   >> TRUE                                                      Auto.
            A
  1<                  >
  2                   |
```

Note that A1 takes on the value -1. Type D to move to A2 and type IF(A1),THEN(5),ELSE(ERROR)<R> which means if A1 is TRUE then give A2 the value 5 otherwise indicate an error:

```
A2                                                              A2
Gen   Enter number or expression                                65504
   >> IF(A1),THEN(5),ELSE(ERROR)                                Auto.
            A
  1                  -1|
  2<                  >
```

```
A2    (IF(A1),THEN(5),ELSE(ERROR))                              A2
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65467
Next:                                                           Auto.
            A
  1                  -1|
  2<                  5>
```

Because A1 was TRUE, A2 has been set to 5. Now change A1 to see the effect on A2. Type U.FALSE<R>:

```
A1    (TRUE)                                                    A2
Gen   Enter number or expression                                65467
   >> FALSE                                                     Auto.
            A
  1<                 -1>
  2                   5|
```

```
A1    (TRUE)                                                    A2
Gen   ERROR called from <A2>                                    65466
  >>                                                            Auto.
              A
  1                       0|
  2                        |
```

A1 now takes the value 0 for FALSE and because of the automatic calculation feature the error message has already come up saying where the error was detected. As a precaution you cannot continue passed the error message until you have pressed the [ESC] key, the current cell is also changed to the cell originating the error so that it can be corrected.

TABLE HANDLING FUNCTIONS

Several functions are available to let you extract values from a specified list of cells. They are used in the same way that we would look up and read values from a table or list.

As a first example you are going to see the LOOKUP function. This function when given a value searches a list for a match to this value and then returns the entry from the adjacent row or column. Consider it as being the same as looking up a value in a printed table where you look for a value in the first column to get your answer from the second, such as logarithmic tables.

A typical example of the use of this function may be finding a commission percentage given sales income. These rates tend to jump from band to band.

```
B16   (LOOKUP(B12,B5...B10))                                              C18
OFin   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                           65009
Next:                                                                     Auto.
                    A                          B               C
  1        COMMISSION CALCULATION|                |               |
  2                              |          Sales |   Commission  |
  3                              |                |      (%)      |
  4                              |                |               |
  5                              |           0.00 |          0.00 |
  6                              |       1,000.00 |          2.50 |
  7                              |       2,000.00 |          5.00 |
  8                              |       4,000.00 |          7.50 |
  9                              |      10,000.00 |         10.00 |
 10                              |      20,000.00 |         20.00 |
 11                              |----------------|               |
 12                Sales achieved |      15,000.00 |               |
 13                              |                |               |
 14                              |            (%) |   Amount paid |
 15                              |================|===============|
 16   Commission one value per band <       10.00 >      1,500.00 |
 17   Commission on a sliding scale |       15.00 |      2,250.00 |
 18                              |================|===============|
```

In this case the salesperson brought in $15000 worth of business and so he managed to get into the band between $10000 and $20000 for which he gets 10% commission.

The form of the function is LOOKUP(value,list), the result returned by the function is taken from the adjacent list.

There is a similar function which you can use in the same way called INTERP which will interpolate a value from a list. It differs from LOOKUP in that the function tries to work out (interpret) the desired answer even if it is present in the list.

```
B17   (INTERP(B12,B5...B10))                                          C18
OFin  ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                       65009
Next:                                                                Auto.
                    A                    B               C
 1       COMMISSION CALCULATION|                |                |
 2                             |         Sales  |   Commission   |
 3                             |                |      (%)       |
 4                             |                |                |
 5                             |          0.00  |         0.00   |
 6                             |      1,000.00  |         2.50   |
 7                             |      2,000.00  |         5.00   |
 8                             |      4,000.00  |         7.50   |
 9                             |     10,000.00  |        10.00   |
10                             |     20,000.00  |        20.00   |
11                             |----------------|                |
12              Sales achieved |     15,000.00  |                |
13                             |                |                |
14                             |           (%)  |   Amount paid  |
15                             |================|================|
16  Commission one value per band |      10.00  |      1,500.00  |
17  Commission on a sliding scale <      15.00  >      2,250.00  |
18                             |================|================|
```

Here the salesperson has been told that the commission will be calculated on a sliding scale based on the sales and commission table. As $15000 worth was sold this is midway between $15000 and $20000 and so he can expect a commission midway between 10% and 20%. The INTERP function does this calculation for you and in this case comes up with the answer 15%.

The CHOOSE function will look at a list and return the value of the cell in the position in the list given by the first argument. The form of this argument is CHOOSE(value,list). The value will be rounded to the nearest whole number if it isn't one already.

```
A8    (CHOOSE(4,A1...A6))                                             A10
Gen   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                       65402
Next:                                                                Auto.
           A
 1                0|
 2             1000|
 3             2000|
 4             4000|
 5            10000|
 6            20000|
 7                 |
 8<            4000>
 9                 |
10                 |
```

In this example the CHOOSE function at A8 has looked through A1...A6 to find the 4th cell and returned the value held in it, in this case 4000.

NPV stands for net present value and is a discounted cash flow function that calculates the effect of a discount rate on a set of cash flow figures. The form of the function is NPV(value,list) where 'value' is the discount rate in percent and the 'list' is a list of cells that contain cash flows. If you do not intend to use THE CRACKER for financial calculations then there is no need to bother following the next example.

```
UNT
B5    (NPV(B4,B2...F2))                                                                F5
OFin  ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                                       65337
Next:                                                                                Auto.
          A              B            C            D            E            F
 1            YEAR|       1984|        1985|        1986|        1987|        1988|
 2       CASH FLOW| 1,000.00 | 1,200.00 | 1,500.00 | 2,000.00 | 1,000.00 |
 3                |           |            |            |            |            |
 4   DISCOUNT RATE|     15.00 |            |            |            |            |
 5   PRESENT VALUE< 4,403.90 >            |            |            |            |
```

In this example you can assume that in 1983 some money is to be invested and the figures above represent the expected yearly returns on that investment. To find out how the investment will perform, the yearly values each need to be converted to 'present values' and summed. In this instance they are all converted to 1983 values.

The first return in 1984 will be calculated as 1000/(1+dr/100). The 1000 is effectively worth less because of the one year taken to get it. The next year 1200 is obtained but this is worth less still because it is discounted once for 1984 and again in 1985 so its present value is calculated as 1200/(1+dr/100)/(1+dr/100) and so on. The value of the return in n years is return/((1+dr/100)^n).

The 'Internal rate of return' is the discount rate necessary to make the present value equal to the initial investment. It can be found by trial and error, changing the value of discount rate until you get the right answer. An example of how they can be automated and the Internal rate of return can be calculated by THE CRACKER is given on the distribution disk and is called IRR.

If you want to do a numerical integration then you would probably use Simpson's rule. You can use the function SIMPRULE to do this directly.

```
B8    (SIMPRULE(PI/8,B1...B5))                                        B8
Gen   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                      65289
Next:                                                               Auto.
              A                   B
 1            SINR(0)|                 0|
 2         SINR(PI/8)| 0.382683432365|
 3         SINR(PI/4)| 0.707106781187|
 4       SINR(3*PI/8)| 0.923879532511|
 5         SINR(PI/2)|                 1|
 6                   |                  |
 7  INTEGRAL OF SINR(X)|                |
 8   BETWEEN 0 AND PI/2< 1.000134584974>
```

In this example 5 values of SINR(X) have been calculated at intervals of PI/8. The SIMPRULE function has been used to obtain an approximate value of the integral, whose exact value is 1. Note that you can use upper or lower case characters in your expressions. The form of the function is SIMPRULE(step,range), where the range must have an odd number of values.

A NOTE ON LISTS

In most functions a list can be specified using a range such as B1...B5. You can however have blank entries in your range and the function will still be worked out correctly. This feature allows you to set up a template worksheet and enter your particular data later. It will also cater for the situation where the number of items will be variable.

AN INTRODUCTION TO 'COMMAND' FUNCTIONS

There are certain functions that we are now going to come across that are fundamentally different to those we have seen so far. The principle difference is that they act on a cell or cells other than the one in which they have been entered, and can be looked on more like commands than like the mathematical functions encountered so far.

These command functions are useful in that they can be used to automate some actions normally entered using direct commands. For example, they can be used to create a worksheet that performs an entire set of calculations as soon as it has been loaded and the automatic recalculation started. Some of the demo files on the distribution disc use this feature. They are also useful for creating subroutines that perform a range of actions or calculations automatically. We will see later how such a technique can be used to create and fill a table of values.

Examples of functions that behave rather like commands are SET and INIT, which assign a value to a distant cell rather than the one they are in, or INC and DEC which can alter the value held in a distant cell.

There are also some command-like functions that may read a value from a distant cell but do not act on anything in particular. An example would be one of the Graphics functions we will see later such as MAINTITLE(crd), which reads the text to be displayed in a graph title from a distant cell, but is unableto display it because of different format types.

Because the command functions act on a cell or cells other than the one in which they have been entered, and because that distant cell may sometimes contain text data, it is sometimes unclear what value the actual function cell itself will display. In fact sometimes you will find they display the same value as held in the distant cell (if they can) and sometimes they will just display a zero.

To avoid confusing the display of the sheet you may wish to place these command functions in a seldom seen area of the sheet, it doesn't matter where they go - THE CRACKER will always find them. Remember that because these are FUNCTIONS with arguments they must always be entered into a cell that has a numeric format.

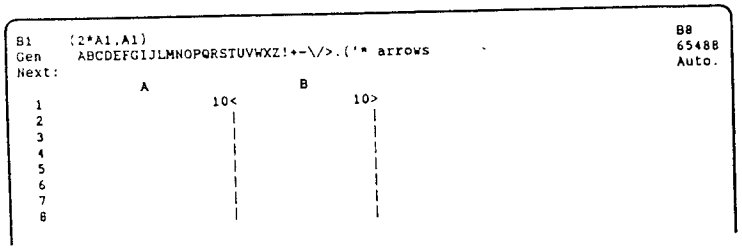
MULTIPLE FUNCTION LINES AND DIVIDING COMMAS

When you are entering functions into a cell it is valid to insert a dividing comma between them. The effect of this dividing comma is to cause an effective restart to the entry as if what follows was the beginning of the line.

The value that will be displayed in the cell will be that of the expression after the last dividing comma.

For example start with a blank sheet and enter 10 into A1 and this expression into B1:

2*A1,A1

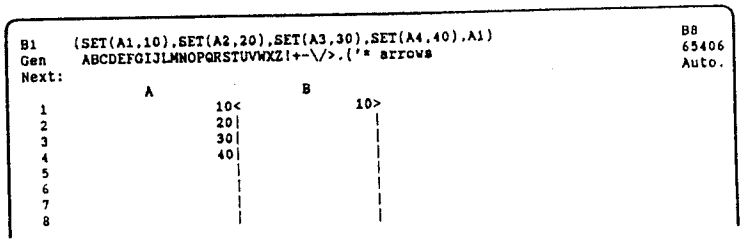


The first calculation, 2*A1, is performed but is effectively forgotten - all that appears in B1 is a copy of A1. This feature so far looks like a waste of time and memory but consider its application to the 'command' functions described above.

Try entering this expression into B1

SET(A1,10),SET(A2,20),SET(A3,30),SET(A4,40),A1

Because the 'command' functions all act on DISTANT cells their effect is performed on the sheet even though cell B1 finishes by only displaying the value obtained after the last comma. None of the earlier commands are wasted. They are also useful with the BLEEP function.



It should be obvious that only these 'command' functions have a useful effect if they are followed by a comma.

Not all command-like functions can be treated in this way and some of the Graphics functions in particular may give unpredictable results if followed by a comma.

DATE AND TIME FUNCTIONS

The DATE and TIME functions have two immediate applications. In financial calculations, such as tax returns or yearly balance sheets, it is often the case that some measure of the time of year has to be taken to determine the required results. By entering values for the time of year automatic adjustments can be made from within the program.

Secondly, there are some functions that are able to keep track of the passage of time since the program was first entered. These are particularly useful when combined with the Input Output control of peripherals when it becomes possible to program THE CRACKER to control equipment such as central heating where the desired output depends on the time of day and year.

DATEAFTER(date,days)
Gives the date that in the number of days specified from the input date. The date must be in the form of a single number, for example 312.1985 - being the 3rd December 1985. Be careful to put months 1 to 9 as 01 to 09.

DAYSAPART(date1,date2)
Gives the number of days between any two specified dates.

DAYOFWK(date)
Returns the day of week as a number. Saturday has a value 0, Sunday 1, Monday 2 etc.

DAYOFYR(date)
Returns the number of days between January 1st and the present day.

The rest of these functions are only available on CP/M PLUS (3) systems. Note for your clock and calendar to be correct you must set them before starting THE CRACKER. Use the utility supplied on your disc called DATE, just type DATE SET at the DOS prompt and follow the screen prompts.

YEAR
Gives the current year.

MONTH
The month.

DAY
The day

HOUR
The hour

MINUTE
The minute

SECOND
The second

DATE
Returns the date in the form of a single number, for example 312.1985 - being the 3rd December 1985.

ZEROTIME
Resets the elapsed time counter. Probably best included in an IF,THEN,ELSE entry.

TIMELAPSE
Returns the elapsed time since the ZEROTIME function was last operative. This is in seconds.

DELAY(n)
Does nothing until n seconds have elapsed. 'n' may be a cell reference or a constant value. For practical reasons make it a cell reference with a value 0 until you actually want to run your application.

I/O FUNCTIONS

These functions can be used to control, or exchange information with, external peripherals. Specific details for their use will depend on the hardware configuration you have. See the manuals for your computer and/or peripherals for details of the port numbers and values to use.

IN(port)
Reads an 8 bit port given by the number or cell reference 'port'.

OUT(port,value)
Outputs a 'value' given by a number or cell reference to the 'port' given by a number or cell reference.

Complex control of external machinery can be acheived by using DO-WHILE loops, the Date - Time functions and the I/O commands.

PRODUCING GRAPHS AND CHARTS

The graphics addition to THE CRACKER has been designed to allow you to generate varying types of graphs from the minimum of input by the user. There are special graphics functions, detailed below, to enable you to specify the data and the format of the graph - this information can be stored anywhere in the spreadsheet.

Note that where 'crd' is specified in the functions below you must put it in the form shown. If you don't, the correct values may not be properly passed to the plotting section of the program. You can't use numerical or expression equivalents except where stated.

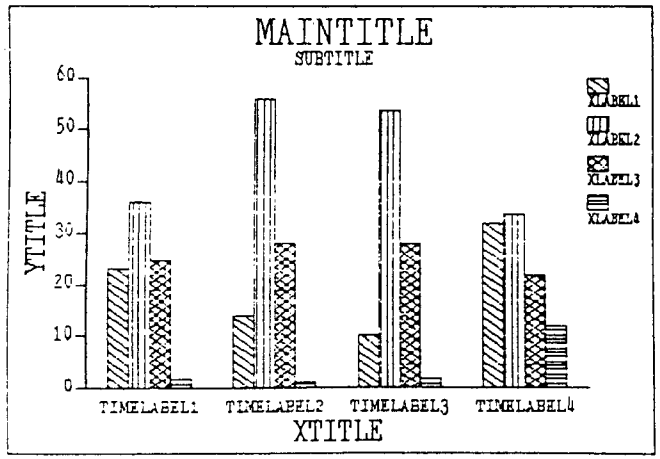
Here are the functions:

TYPEPLOT(crd)
The type of graph or chart plot you want. Give a value between 1 and 18 in the cell referred to by this function.

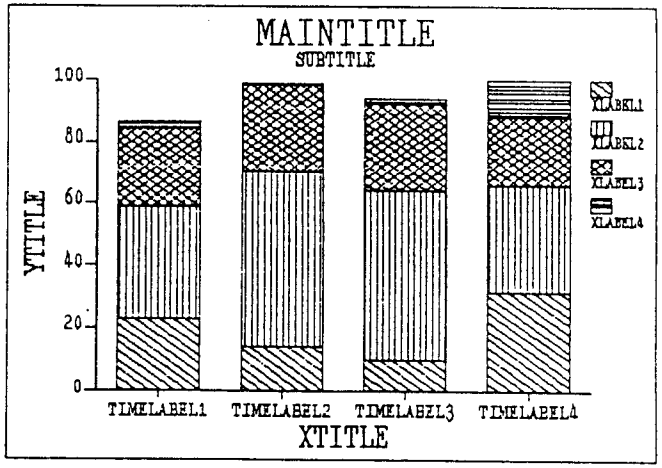
The plot types are detailed below:-

Business (these graphs require timelabels rather than actual Xvalues:

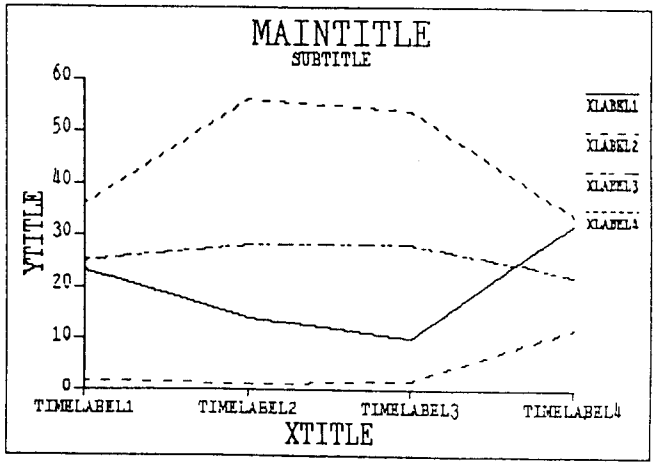
1. Bar chart (histogram) - there may be more than one bar over each timelabel



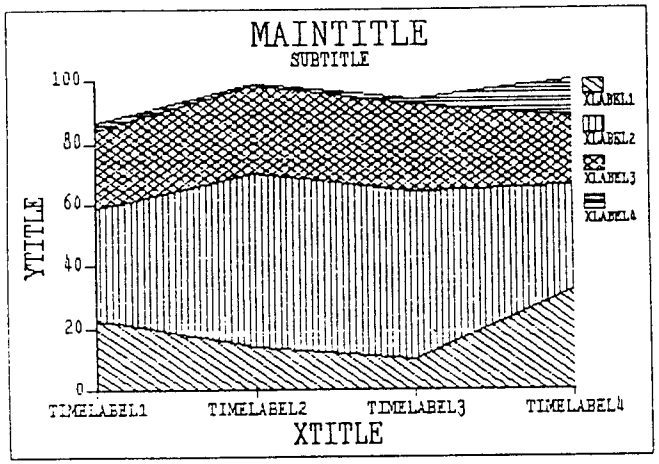
2. Stacked bar chart - each bar for each timelabel is superimposed over the others so that only the overlap is shown



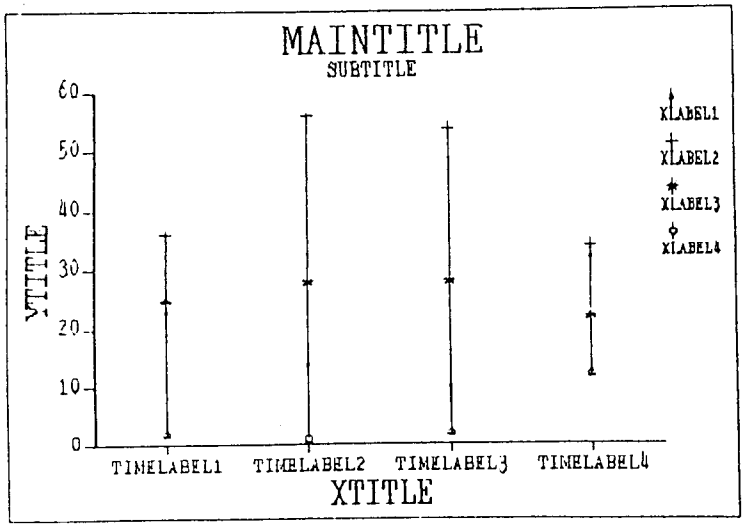
3. Line chart - lines join each point on each of the plot categories



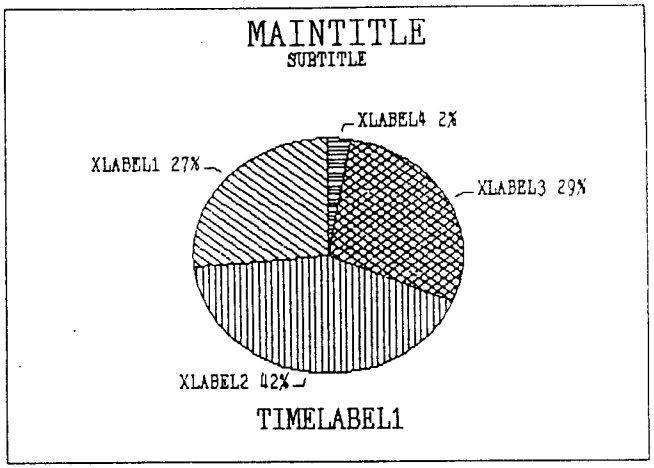
4. Area chart - as above except that each enclosed area is hatched



5. Hi-lo chart - each of the plot categories are shown as a marker above each timelabel. The markers are joined by a vertical line.

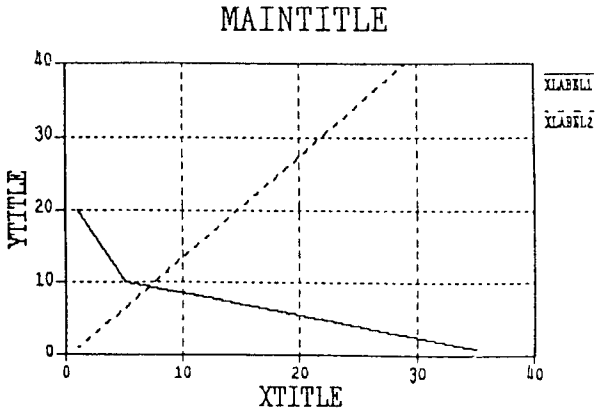


6. Pie chart - the familiar segmented circular chart.



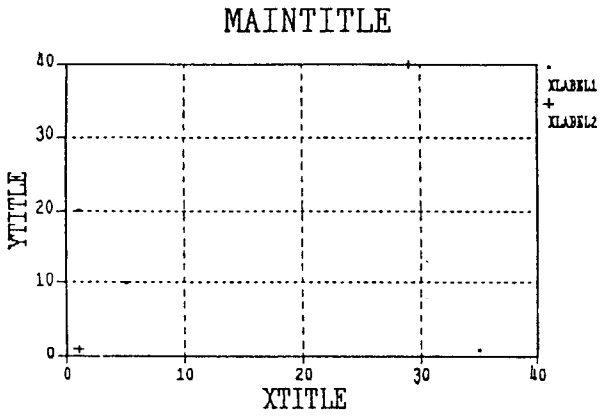
Statistical, engineering and scientific:

7. X:Y line joining points



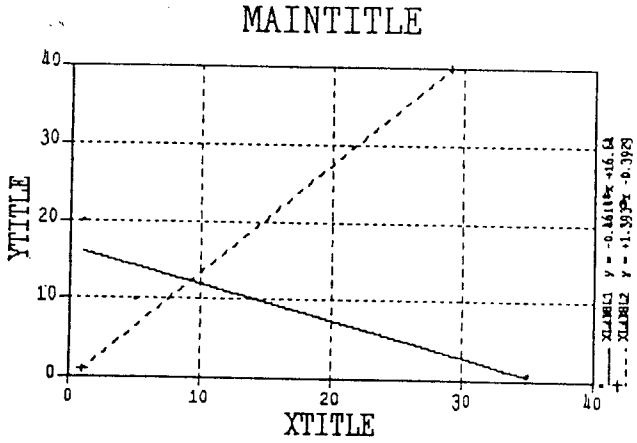
8. LogX:Y line joining points
9. X:LogY line joining points
10. LogX:LogY line joining points

11. X:Y points only



12. LogX:Y points only
13. X:LogY points only
10. LogX:LogY points only

15. X:Y points and best fit line



16. LogX:Y points and best fit line
17. X:LogY points and best fit line
18. LogX:LogY points and best fit line

The best fit lines of plot types 15...18 are obtained by the least squares method.

The other graphics functions are:

MAINTITLE(crd)
The cell coordinate is a pointer to a text cell where the main title is to be found.

SUBTITLE(crd)
The same idea as for the maintitle applies to the subtitle.

YTITLE(crd)
The coordinate pointer is to the title up the Y axis on the left hand side.

XTITLE(crd)
Where coordinate refers to where the X axis title is. This is the one at the bottom of the graph.

XLABEL(crd...crd), or XLABEL(crd,crd,crd,etc.)
This points to the X labels which are the legend box items on the right explaining what the plot lines etc. refer to. THE CRACKER also uses this function to discover how many plot lines you want to display.

Unlike the other functions in this list the range can also be a list of individual items each one pointing to one of the X labels. Please make an effort to get the number of X labels correct as the program counts them to see how many lines or groups of items there are. If for example you get it wrong and enter too many Xlabels you will be be informed there are Y values missing because it is expecting to have to draw further lines.

TIMELABEL(crd...crd)
These are the tagged items on the X or bottom axis that show what exactly is being plotted. They are not always time labels but it is very common in financial graphs for example to plot values according to month or year. This option refers to plot types 1...6 only, the other graph types will expect actual numerical values on the lower line.

There are some restrictions on the maximum size of the graph labels. consult the Command Reference for more details.

YVALUE(n,crd...crd)
This function points to the actual Y values to be plotted. The 'n' refers the number of the plotting line to which you are referring. There will be one of these functions for each line.

YMAXIMUM(crd)
The 'crd' in this case can be a value or a reference to a value that specifies the maximum figure to be shown on the Y axis. Your choice will be rounded to a suitable nearby value to improve the presentation.

YMINIMUM(crd)
This is a similar function to the above for a minimum value.

Note that because THE CRACKER rounds the value for the minimum to the best nearby value to give an attractive display you may have to experiment and perhaps set the YMINIMUM value lower than you first thought in order to get the desired display.

XVALUE(n,crd...crd)
This function points to the actual X values to be plotted. The 'n' refers the number of the line to which you are referring. There will be one of these functions for each line. (plot types 7...18 only, use TIMELABELS for 1...6)

XMAXIMUM(crd)
The crd may be a value or a reference to a value that specifies the maximum value to be shown on the X axis. Your choice will be rounded to a suitable nearby value to improve the presentation. (plot types 7...18)

XMINIMUM(crd)
Similarly for a minimum value.

See the notes given above for YMINIMUM.

The XMINIMUM command only works on plot types 7 or above.

Remember that because the graphics commands are entered as FUNCTIONS followed by values or coordinates each of the above should be placed into a cell that has been given a numerical format, even though the information they refer to may be text. The text itself of course has to be placed in cells of a text format.

You may be wondering what each of these cells that contain the above functions will actually display on the screen. In the majority of the cases the display will show zero. Some of the functions will display a numeric value if it is felt to be useful. For example the cell that contains the XLABELS function will show the number of Xlabels defined. The cell that contains TYPEPLOT will show the number of the graph type chosen.

There is an example of a business plot spreadsheet on the file **PLOT1.MEM** and one for scientific work on **PLOT2.MEM**. Load these spreadsheet and use the eXchange rules/formulae command to the how the functions are used in practice. The HELP utility will give you reminders about each of these functions and plot types.

MAKING A GRAPH IN PRACTICE

Business type

Prepare a sheet with the data you want plotted on it.The actual data values can be up and down columns or across lines, as long as they lie in a range of coordinates. It is not critical how the other items are laid out so just set aside an area to put in the plotting instructions. Each of these instructions is a pointer to where the particular data items you want displayed are to be found.

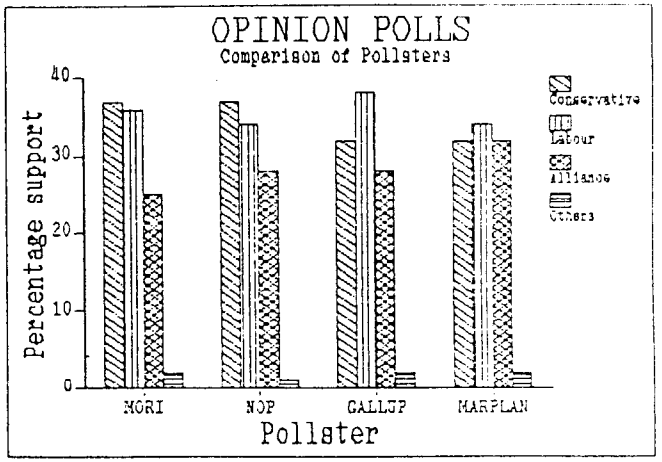
Start by putting in the TYPEPLOT function and then the pointers to the various titles, although these are optional. These are MAINTITLE, SUBTITLE, YTITLE and XTITLE.

Next show where the TIMELABELs text can be found. Remember these are typically the months or years that go across the bottom of the page, they don't in fact have to relate to time but usually do.

Lastly point to the YVALUEs, which are the actual sets of data. There will be one YVALUE entry for each set.

If you particularly wish to specify the maximum or minimum values you want plotted then use the YMAXIMUM and YMINIMUM functions. Normally don't use these functions as the program will work out all the maximums and minimums for you automatically.

```
A6                                                                  E21
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.['* arrows                     52341
Next:                                                               Auto.
          A                    B          C          D          E
  1       (1)                  |          |          |          |
  2 OPINION POLLS              |          |          |          |
  3 Comparison of Pollsters    |          |          |          |
  4 Percentage support         |          |          |          |
  5 Pollster                   |          |          |          |
  6<                          >|Conservative|Labour  |Alliance  |Others
  7 MORI                       |       (37) |    (36) |    (25) |    (2)
  8 NOP                        |       (37) |    (34) |    (28) |    (1)
  9 GALLUP                     |       (32) |    (38) |    (28) |    (2)
 10 MARPLAN                    |       (32) |    (34) |    (32) |    (2)
 11  <11>  (typeplot(A1))      |            |         |         |
 12  <10>  (maintitle(A2))     |            |         |         |
 13   <9>  (subtitle(A3))      |            |         |         |
 14   <8>  (ytitle(A4))        |            |         |         |
 15   <7>  (xtitle(A5))        |            |         |         |
 16   <6>  (xlabel(B6...E6))   |            |         |         |
 17   <5>  (timelabel(A7...A1|            |         |         |
 18   <4>  (yvalue(1,B7...B10|            |         |         |
 19   <3>  (yvalue(2,C7...C10|            |         |         |
 20   <2>  (yvalue(3,D7...D10|            |         |         |
 21   <1>  (yvalue(4,E7...E10|            |         |         |
```

Similar instructions apply for plot types 7...18, scientific graphs. In these cases XVALUES replace TIMELABELs.

If any errors are encountered while using the graphics an error message will be given and a return will be made to the spreadsheet. This is an early version of the graphics program and so if you find you can not return to the main program don't worry as your work to date will be found on the file SECURITY.MEM (Z80 versions only, 16 bit versions are not overlaid).

If you do any EDITING of the plotting instructions or make any changes that do not force a recalculation, you may end up with an error message or values that do not seem to be true. This is most likely when you use direct values in your functions rather than cell coordinates.

An example may be

YMAXIMUM(20000)

and a case where it would not occur would be

YMAXIMUM(B7)

Don't worry about this, just remember the safest way to handle plotting functions that do not contain cell references is to overwrite them rather than edit them.

The instructions for plotting are updated when a recalculation is done so if you have made changes the latest instructions may not have been passed causing an error. If you get such an error message just use the ! FORCE RECALCULATION COMMAND and try the plot again.

THE CRACKER TUTORIAL
VI. AUTOMATING DATA MANIPULATION

SOME MORE ADVANCED FEATURES OF THE WORKSHEET

USING MACRO COMMAND GROUPS

Often you will want to go through the same set of commands repeatedly. A typical example is the changing of the format of all the cells in one column which can be both time consuming and tedious. To get round this you can use the *MACRO command. A macro is a computing term that you will come across in many programs, and it is defined as a linked set of commands that can be set in operation by issuing a single instruction. It is really only a shortcut to save you from a lot of unnecessary typing.

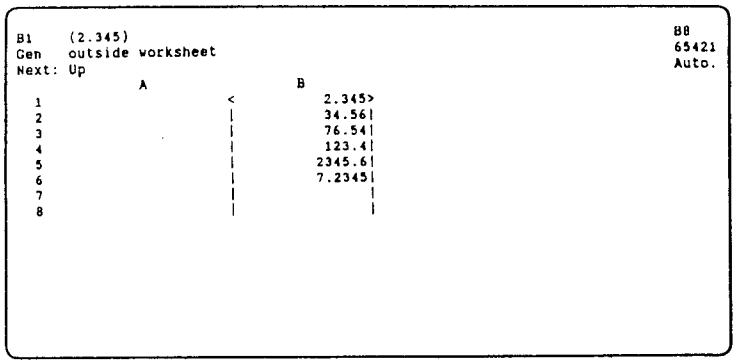
THE CRACKER will let you create predefined sets of linked commands, by entering the command letters into a cell as if you were actually typing them in to be acted on immediately, with the exception that an '@' symbol is inserted in place of a carriage return.

The macro is executed by typing '*' followed by the coordinate reference to the cell in which it is stored. For example, *A1 executes the macro stored in cell A1.

The cell that contains the commands should be set to any text format. The maximum size of a macro is limited but you can put your macro in more than one cell by finishing with a reference to the continuation macro. For example, you might finish your A1 macro with a reference *A2 to force it to continue with the commands entered in A2.

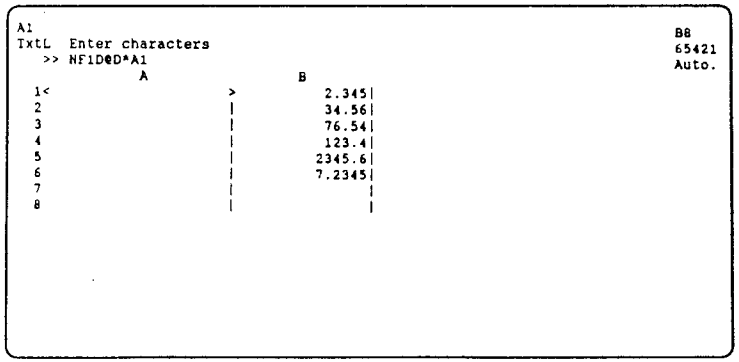
If you want your macro to loop and be carried out repeatedly then finish it with a reference to itself. For example with cell A1 this would mean finishing with *A1. Don't worry about this causing an endless loop, there are lots of ways to make the macro come to an end.

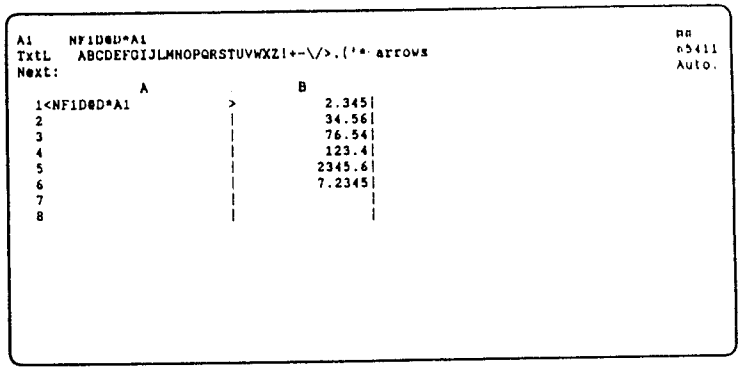
As an illustration the following example takes a column of numbers in general format and changes them all to one place decimal format.



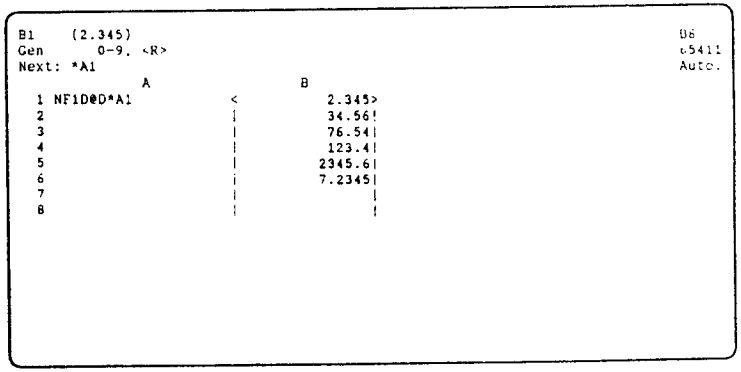
Start by moving the cursor to cell A1 where the macro is to be entered. Type '.' to get into entry mode and then type NF1D@D*A1 which says 'new format one place decimal, carriage return, cursor down and finally do macro A1 again'.

The @ symbol stands for the <R> carriage return.





Next move the cursor to the location you want the macro to start its operations. In this case it is cell B1, and then type *A1 to start the macro going.



The commands will then work their way down the column.

```
B2    (34.56)                                                    B8
Gen                                  <R>                        65411
Next: New Format to be 1 places Decimal                         Auto.
              A                    B
  1 NF1D@D*A1          |            2.3|
  2                    <           34.56>
  3                    |           76.54|
  4                    |          123.4|
  5                    |         2345.6|
  6                    |         7.2345|
  7                    |               |
  8                    |               |
```

In the screen above the macro has got as far as cell B2.

```
B6    (7.2345)                                                   B6
1Dec  outside worksheet                                         65411
Next: Down                                                      Auto.
              A                    B
  1 NF1D@D*A1          |            2.3|
  2                    |           34.6|
  3                    |           76.5|
  4                    |          123.4|
  5                    |         2345.6|
  6                    <            7.2>
```

A macro is brought to a stop by any error message. In the screen above you can see that the D command for down cannot be carried out past cell B15 because that is the end of the worksheet. Just type [ESC] and your looping macro operations will be completed and you will be in a position to go onto your next command.

```
B6    (7.2345)                                                   B6
1Dec   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65411
Next:                                                           Auto.
              A                    B
  1 NF1D@D*A1          |            2.3|
  2                    |           34.6|
  3                    |           76.5|
  4                    |          123.4|
  5                    |         2345.6|
  6                    <            7.2>
```

A MACRO differs from a subroutine function, which you will learn about next, in that it is a list of direct commands rather than functions - the MACRO can contain a command such as 'NF' for New Format, which is something that cannot normally be entered into a cell as part of an expression. The subroutine is another type of loop, one that performs normal expression functions repeatedly.

It is of incidental interest that if you were to restrict your MACRO definitions to the first 9 cells of column A they could be accessed by just typing *1,*2 etc. This is a feature of older versions of THE CRACKER that has been retained for compatibility with existing data sets.

FUNCTIONS THAT ALLOW LOOPING

Sometimes it would be very useful if you could use a few formulae repetitively to work towards an answer. For instance, when working out the internal rate of return example, a short entry has been set up that tries a range of possibilities and stops at the nearest.

The functions that allow you to do this repetitive calculations are often referred to as loops. THE CRACKER provides you with two functions, DO and WHILE, designed to make the setting up of such loops very easy. Unlike the IF THEN example seen earlier, the command word DO is not followed directly by an expression but rather by a reference to a range of cells that contains the desired expression(s).

Note that you must specify a RANGE so your working must be in one line or one column and not in a block. However the function looks at the range and finds the highest and lowest order of calculation number and then all those entries between the numbers are recalculated no matter where they are in the sheet. It does not matter if some of your intermediate calculation numbers are not in the specified range as they will still be correctly calculated.

To use a loop enter the DO (range) function which performs the desired calculation followed by a comma then any other expression or function that you wish. Usually this second function will change a value somewhere to act as a loop counter (counts the number of times that the calculation has been done).

There are several related functions that make setting up a counter easy for you namely INIT, SET, INC and DEC. These respectively initialize a cell entry, set a value to to a cell entry, increment a value (increase by one) and decrement a value (decrease by one).

After the loop counter section you should type in a further comma and then use the WHILE function. This function has a logical argument such as a logical expression finding that determines whether the loop counter has reached a certain value, often zero. If the WHILE function is TRUE then the cell formula is started again at the DO and the WHILE tested repeatedly it becomes FALSE.

This description may sound a little complicated, but it will become clearer if you follow through the examples and then re-read this section.

Firstly, we will examine the SET function in more detail. The command takes the form: SET(crd,value). The value can be either a constant or an expression, and it may even include the referenced coordinate. Consider the expression SET(A2,A2+1) which has the same affect as an INC(A2) command.

The SET command works in a similar way to entering a constant value into a cell using the entry command, except that the value is entered as a result of the expression rather than directly. It is therefore a useful way of automating the entry of cell constants and ensuring that the value will be updated if the referred cell values change.

```
A1                                                          A2
Gen   Enter number or expression                            65519
   >> SET(A2,5)                                             Auto.
         A
  1<              >
  2              5|
```

```
A1    (SET(A2,5))                                           A2
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows             65485
Next:                                                       Auto.
         A
  1<             5>
  2              5|
```

SET serves to set a value into one cell while you are entering an expression into another. Note that the destination cell is set even before the expression is fully entered. If you press enter immediately then the cell that contains the set function would itself take on the value shown in the distant cell that has been set.

The INC function increments the value stored at a particular coordinate reference. It is used most effectively in conjunction with a DO WHILE loop when it forms part of a counter to control the loop. Here again the function takes effect before the expression is completed, so as you enter such a formula you will see A2 first have the value 5 and then 6 as it is incremented.

```
A1    (SET(A2,5))                                           A2
Gen   Enter number or expression                            65485
   >> SET(A2,5),INC(A2)                                     Auto.
         A
  1<             5>
  2              6|
```



```
A1    (SET(A2,5),INC(A2))                                   A2
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows             65477
Next:                                                       Auto.
         A
  1<             6>
  2              6|
```

Now, although you are manipulating the values that are held in cell A2 don't forget that your current cursor position is still A1 and that it is into A1 that you are entering your expression.

Remember that A1, i.e. the current cursor position, will take on a value that is calculated from the expression or function after the last dividing comma in your expression list. You can, therefore, put a further comma in followed by an expression or value that doesn't manipulate A2, that will be entered into A1.

```
A1    (SET(A2,5),INC(A2))                                   A2
Gen   Enter number or expression                            65477
   >> SET(A2,5),INC(A2),2*A2                                Auto.
         A
  1<             6>
  2              6|
```

```
A1    (SET(A2,5),INC(A2),2*A2)                              A2
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows             65472
Next:                                                       Auto.
         A
  1<            12>
  2              6|
```

In this case the entry at A1 has ended up with the value 2*A2 which is 12. The cell entries as a whole have become a series of instructions executed in turn.

SUBROUTINES USING THE DO FUNCTION

As well as being a part of a loop the DO function can be used without the WHILE to act as a subroutine that is only executed once every time it is called.

You may be familiar with subroutines but if not here is an explanation.

A subroutine is a group of formulae which you may want to use repeatedly. Rather than enter the formulae many times over you simply have to type them in once and access them with the DO function. You will see the similarity here with a macro command. Every time the subroutine function is found in a cell during recalculation the whole of the group is calculated again. Now have a look at it in practice:

```
A3    (2*A4)                                                    A4
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65492
Next:                                                           Auto.
          A
  1                |
  2                |
  3<              6>
  4               3|
```

Here is a small group, in this case just 2 cells. The formula is also very simple. You will see how the DO function is used to find out what 2 times any number is not just the 3 above:

```
A1                                                              A4
Gen   Enter number or expression                                65492
   >> SET(A4,5)                                                 Auto.
          A
  1<               >
  2                |
  3               6|
  4               5|
```

First you use the SET function to place the number you want doubled into A4. Note that the display for cell A4 has already changed before you have finished typing:


```
A1                                                              A4
Gen   Enter number or expression                                65492
   >> SET(A4,5),DO(A3...A4)                                     Auto.
          A
  1<               >
  2                |
  3               6|
  4               5|
```

Next you put in a comma and the DO function, the argument of which MUST be a range and not an individual entry (although it can be just two cells, one of which is blank) or a block. This is not as restrictive as it may sound because the DO function will calculate a complete sequence of formulae. Each cell in the range you specify will have an order of calculation number. The DO function will find the one that has the lowest number and the one that has the highest. As each function is calculated in turn every cell in the sheet that falls within the bounds of these two calculation numbers will also be automatically recalculated. This means you will get the required results provided that your range includes both the start and end expressions for the cells you wish to effect.

```
A1    (SET(A4,5),DO(A3...A4))                                   A4
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65458
Next:                                                           Auto.
          A
  1<              10>
  2                |
  3               10|
  4                5|
```

When you complete the expression the calculation is carried out. If the answer you want has not appeared in cell A1 it can be made to do so by following the expression by a comma and a reference to the cell that does have the answer, in other words by effectively resetting the expression you have typed and by copying the value of the answer cell into the current cell.

Now you can try using the subroutine group we have set up to multiply 7 by 2. To illustrate that you can call the subroutine from anywhere in the sheet you can do this in cell A2:

```
A2                                                              A4
Gen   Enter number or expression                                65458
   >> SET(A4,7),DO(A3...A4)                                     Auto.
            A
  1              10|
  2<               >
  3              10|
  4               7|
```

```
A2    (SET(A4,7),DO(A3...A4))                                   A4
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65426
Next:                                                           Auto.
            A
  1                  10|
  2<                 14>
  3                  14|
  4                   7|
```

Note that the cells that contain the subroutine change as well. Thus you end up with 14 in cell A2 as well as the 10 in A1. Because of this it is as well to keep the subroutine cells well out of the way of the rest of the calculations. The group used for subroutine calculation should not be used or referred to by the main body of the worksheet other than as a subroutine call or it may give nonsense values. You are, after all, changing the constants in it several times during the calculation.

You can change the cell values with SET commands before each DO function.

This is a trivial example but the same technique can be used for much larger sections of the worksheet.

LOOPING USING FUNCTIONS DO AND WHILE

Now we are going to look at how to use the DO and WHILE functions together to build up a repeating loop. The first task is to set aside a cell to act as a counter to keep track on how many times the loop has been performed. Do this with the INIT function.

The INIT function is almost identical to SET but the target cell will be given a very low order of calculation number and hence you can be sure it will be re-calculated early on in every forced recalculation. This is important because every time you run the loop INIT makes sure that the counter is reset to its original value. Using SET in these circumstances may mean that the counter cell increases in value after the loop is used once and the condition tested for by the WHILE function may never be met.

```
A1                                                              A5
Gen   Enter number or expression                                65519
   >> INIT(A2,0)                                                Auto.
            A
  1<                 >
  2                 0|
  3                  |
  4                  |
  5                  |
```

```
A1    (INIT(A2,0))                                              A5
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65484
Next:                                                           Auto.
            A
  1<                 0>
  2                  0|
  3                   |
  4                   |
  5                   |
```

In this case the counter cell is A2. The object of this example is to multiply the value of the counter by 2 and then increment the value and do it again. A criterion for stopping will be specified.

```
A5                                                              A5
Gen   Enter number or expression                                65464
   >> A2                                                        Auto.
            A
  1                  0|
  2                  0|
  3                   |
  4                   |
  5<                  >
```

A reference to the counter is placed in A5:

```
A5    (A2)                                                    A5
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows               65471
Next:                                                        Auto.
              A
  1                    0|
  2                    0|
  3                     |
  4                     |
  5<                   0>
```

At this stage it has the value 0.

```
A4                                                           A5
Gen   Enter number or expression                            65471
   >> 2*A5                                                  Auto.
              A
  1                    0|
  2                    0|
  3                     |
  4<                    >
  5                    0|
```

Now the cursor moves to A4 and the formula for multiplying by 2 is entered. The counter and formula are now set up so the loop can be established:

```
A3                                                           A5
Gen                                                         65456
   >> DO(A4...A5),INC(A2)                                   Auto.
              A
  1                    0|
  2                    1|
  3<                    >
  4                    0|
  5                    0|
```

So far the formaula is similar to the subroutine example:

```
A3                                                           A5
Gen                                                         65456
   >> DO(A4...A5),INC(A2),WHILE(A2<10)                      Auto.
              A
  1                    0|
  2                    2|
  3<                    >
  4                    2|
  5                    1|
```


The heart of the loop is the WHILE function which has the special property that if the logical argument that follows it is TRUE then the expression is wound back to the DO and repeated. In the example it remains TRUE while A2 is less than 10. But note that after the DO function the counter A2 is incremented. The multiply by 2 formula therefore has a different start value. Overall the effect is that on the screen you see A5 going from 0 to 9 while A4 goes from 0 to 18. Not a lot of use in this case but it shows the loop working 10 times. Later you will see the looping used to fill a table.

Reminder: To make it easier to see the rules of the spreadsheet you can convert the display to show just the formulae. Let us try this now. First the column width needs to be increased:

```
A3    (DO(A4...A5),INC(A2),WHILE(A2<10))                     A5
Gen                          0-9, <R>                       65413
Next: New width of column 45                                Auto.
              A
  1                    0|
  2                   10|
  3<                   0>
  4                   18|
  5                    9|
```

Then the X for eXchange command is used:

```
A3    (DO(A4...A5),INC(A2),WHILE(A2<10))                     A5
Gen                          <R>                            65413
Next: eXchange rules/results                                Auto.
                          A
  1                                          0|
  2                                         10|
  3<                                         0>
  4                                         18|
  5                                          9|
```

```
A3    (DO(A4...A5),INC(A2),WHILE(A2<10))                     A5
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows              52691
Next:                                                        Auto.
                          A
  1   <1>  (INIT(A2,0))                          |
  2        (set)                                 |
  3<  <4>  (DO(A4...A5),INC(A2),WHILE(A2<10))    >
  4   <3>  (2*A5)                                |
  5   <2>  (A2)                                  |
```

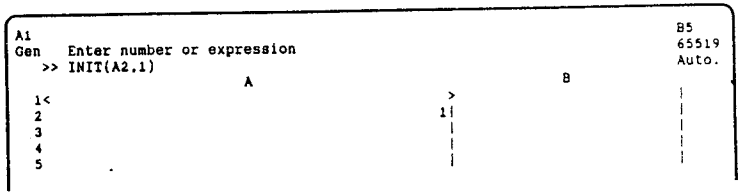
You can now see all the formulae. The number in front of the formula is the order of recalculation number.

The way the looping is organised means that the DO formulae will always be calculated once before the WHILE is tested. This ties in with the way DO and WHILE work in the main computer languages. If this is not the way you would like the loop to work you can precede it with an ordinary IF THEN logic test.

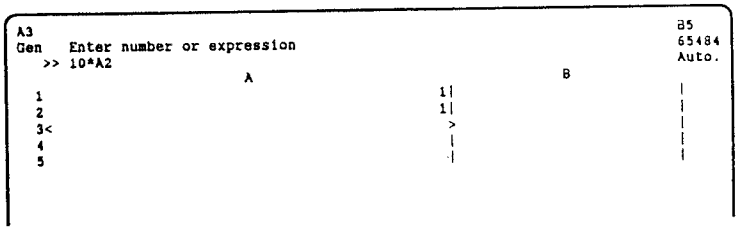
TABLE FILLING USING THE DO FUNCTION

The SET command can be used for generating tables within the speadsheet. As an example, we shall consider the generation of a table of values for sin(x) between 10 and 90 degrees.

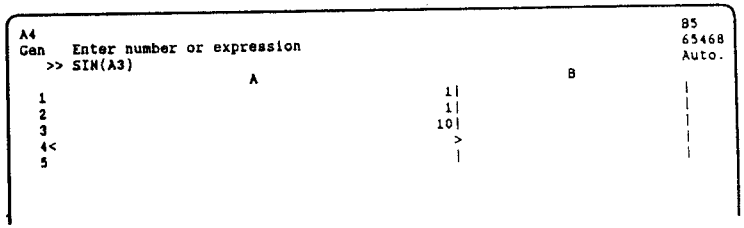
As before, start by initializing a counter to keep track of how far into the table you have got - in this case cell A2 is set up with 1:



As we require the sine values every 10 degrees you can use the counter multiplied by 10. The following formula is inserted in cell A3:



Now you have a value for degrees you can refer to it with a SIN function in cell A4:



The object is to fill a table in column B with sine values at 10 degree intervals. In order to transmit the calculated value to the required cell you can use the SET function together with the CRD(col,line) function rather than giving a specific coordinate reference.

The CRD function can be used anywhere you would use a coordinate reference when typing directly, and it is set to a cell reference depending on the values of the two arguments.

```
A5                                                              B7
Gen   Enter number or expression                                65450
   >> SET(CRD(2,A2),A4)                                         Auto.
                    A                          B
  1                                1|  0.17364817766693|
  2                                1|                   |
  3                               10|                   |
  4          0.17364817766693|                   |
  5<                               >|                   |
  6                                 |                   |
  7                                 |                   |
```

The CRD function is of the form CRD(column,line). In the case above the column number is 2 which is column B. The line number is the value of A2 which is 1. The CRD function will therefore become the cell reference in B1 and will behave exactly as if the entry read SET(B1,A4). As you can see the cell B1 has already been set to the value of A4.

A2 is a counter and can be made to change in value by using a DO function. As A2 changes so will the CRD function become different cell references. In this way we can fill a column of cells with values and form a table.

The next step is for you to enter the DO - WHILE loop:

```
A6                                                              B7
Gen   Enter number or expression                                65408
   >> DO(A3...A5),INC(A2),WHILE(A2<10)                          Auto.
                    A                          B
  1                                1|  0.17364817766693|
  2                                2|                   |
  3                               10|                   |
  4          0.17364817766693|                   |
  5          0.17364817766693|                   |
  6<                               >|                   |
  7                                 |                   |
```


This says 'calculate the sine values as set out in cells A3...A5 then increase the counter by 1 and do it again as long as the counter is less than or equal to 9':

```
A6                                                              B7
Gen   Calculating                                               65365
   >>                                                           Auto.
                    A                          B
  1                                1|  0.17364817766693|
  2                                5|  0.3420201433256691
  3                               50|                0.5|
  4          0.7660444431189781  0.6427876096865391
  5          0.6427876096865391  0.7660444431189781
  6                                 |                   |
  7                                 |                   |
```

The above screen shows the display after the first five sine values have been calculated.

```
A6    (DO(A3...A5),INC(A2),WHILE(A2<10))                        B10
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 65253
Next:                                                           Auto.
                    A                          B
  1                                1|  0.17364817766693|
  2                               10|  0.3420201433256691
  3                               90|                0.5|
  4                                1|  0.6427876096865391
  5                                1|  0.7660444431189781
  6<                               0>  0.8660254037844391
  7                                 |  0.9396926207859091
  8                                 |  0.9848077530122081
  9                                 |                1|
 10                                 |                   |
```

The calculation has now been completed and below you can see the full set of formulae used. Remember to get this display you use the X command:

```
A6    (DO(A3...A5),INC(A2),WHILE(A2<10))                        B10
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                 52591
Next:                                                           Auto.
                    A                                  B
  1   <1>  (INIT(A2,1))                    |  (set)           |
  2        (set)                           |  (set)           |
  3   <2>  (10*A2)                         |  (set)           |
  4   <3>  (SIN(A3))                       |  (set)           |
  5   <4>  (SET(CRD(2,A2),A4))             |  (set)           |
  6<  <5>  (DO(A3...A5),INC(A2),WHILE(A2<10))  >  (set)        |
  7                                        |  (set)           |
  8                                        |  (set)           |
  9                                        |  (set)           |
 10                                        |                  |
```

You can add to the sophistication of the display by actually having the degrees presented as well. To do this you must first insert another column and a line at line 6.

```
A6                                                                  C12
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     52591
Next:                                                               Auto.
                    A                          B          C
  1   <1>  (INIT(A2,1))                    |     |     (set)     |
  2        (set)                           |     |     (set)     |
  3   <2>  (10*A2)                         |     |     (set)     |
  4   <3>  (SIN(A3))                       |     |     (set)     |
  5   <4>  (SET(CRD(2,A2),A4))             |     |     (set)     |
  6<                                       >     |               |
  7   <5>  (DO(A3...A5),INC(A2),WHILE(A2<10))  |     |     (set)     |
  8                                        |     |     (set)     |
  9                                        |     |     (set)     |
 10                                        |     |     (set)     |
 11                                        |     |               |
 12                                        |     |               |
```

With the new column B inserted you can see that the CRD function in cell A5 needs adjusting to refer to column C. This is NOT done automatically for you and so you will have to re-enter it or use the Edit command to change the first 2 to a 3. For the practice let us do the latter.

```
A5    (SET(CRD(2,A2),A4))                                           C12
Gen         <R>                                                     52591
Next: Edit                                                          Auto.
                    A                          B          C
  1   <1>  (INIT(A2,1))                    |     |     (set)     |
  2        (set)                           |     |     (set)     |
  3   <2>  (10*A2)                         |     |     (set)     |
  4   <3>  (SIN(A3))                       |     |     (set)     |
  5<  <4>  (SET(CRD(2,A2),A4))             >     |     (set)     |
  6                                        |     |               |
  7   <5>  (DO(A3...A5),INC(A2),WHILE(A2<10))  |     |     (set)     |
  8                                        |     |     (set)     |
  9                                        |     |     (set)     |
 10                                        |     |     (set)     |
 11                                        |     |               |
 12                                        |     |               |
```



```
A5    (SET(CRD(2,A2),A4))                                           C12
Gen         "Xchange" char,<R>                                      52591
Edit: SET(CRD(3,A2),A4)                                             Auto.
                    A                          B          C
  1   <1>  (INIT(A2,1))                    |     |     (set)     |
  2        (set)                           |     |     (set)     |
  3   <2>  (10*A2)                         |     |     (set)     |
  4   <3>  (SIN(A3))                       |     |     (set)     |
  5<  <4>  (SET(CRD(2,A2),A4))             >     |     (set)     |
  6                                        |     |               |
  7   <5>  (DO(A3...A5),INC(A2),WHILE(A2<10))  |     |     (set)     |
  8                                        |     |     (set)     |
  9                                        |     |     (set)     |
 10                                        |     |     (set)     |
 11                                        |     |               |
 12                                        |     |               |
```

You move to the relevant position using the space bar and then type an X for eXchange. Type the 3 and then press the <R> twice, once to get out of the eXchange mode and the second to get out of the edit mode. This editing will force a recalculation once completed:

```
A5    (SET(CRD(3,A2),A4))                                           C12
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     52591
Next:                                                               Auto.
                    A                          B          C
  1   <1>  (INIT(A2,1))                    |     |     (set)     |
  2        (set)                           |     |     (set)     |
  3   <2>  (10*A2)                         |     |     (set)     |
  4   <3>  (SIN(A3))                       |     |     (set)     |
  5<  <4>  (SET(CRD(3,A2),A4))             >     |     (set)     |
  6                                        |     |     (set)     |
  7   <5>  (DO(A3...A5),INC(A2),WHILE(A2<10))  |     |     (set)     |
  8                                        |     |     (set)     |
  9                                        |     |     (set)     |
 10                                        |     |     (set)     |
 11                                        |     |               |
 12                                        |     |               |
```

Now you can enter the extra SET function to fill up column B with the degree values which are calculated in cell A3:

```
A6                                                                          C12
Gen   Enter number or expression                                            5259i
  >>  SET(CRD(2,A2),A3)                                                     Auto.
                       A                                  B         C
 1   <1>   (INIT(A2,1))                              |    |       (set)      |
 2         (set)                                     |    |       (set)      |
 3   <2>   (10*A2)                                   |    |       (set)      |
 4   <3>   (SIN(A3))                                 |    |       (set)      |
 5   <4>   (SET(CRD(3,A2),A4))                       |    |       (set)      |
 6<                                                  >    |       (set)      |
 7   <5>   (DO(A3...A5),INC(A2),WHILE(A2<10))        |    |       (set)      |
 8                                                   |    |       (set)      |
 9                                                   |    |       (set)      |
10                                                   |    |                  |
11                                                   |    |                  |
12                                                   |    |                  |
```

```
A6     (SET(CRD(2,A2),A3))                                                  C12
Gen     ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                            5249i
Next:                                                                       Auto.
                       A                                  B         C
 1   <1>   (INIT(A2,1))                              |    |       (set)      |
 2         (set)                                     |    |       (set)      |
 3   <2>   (10*A2)                                   |    |       (set)      |
 4   <4>   (SIN(A3))                                 |    |       (set)      |
 5   <5>   (SET(CRD(3,A2),A4))                       |    |       (set)      |
 6<  <3>   (SET(CRD(2,A2),A3))                       >    |       (set)      |
 7   <6>   (DO(A3...A5),INC(A2),WHILE(A2<10))        |    |       (set)      |
 8                                                   |    |       (set)      |
 9                                                   |    |       (set)      |
10                                                   |    |                  |
11                                                   |    |                  |
12                                                   |    |                  |
```


This will also force a recalculation to give the tables you want:

```
A6     (SET(CRD(2,A2),A3))                                                  C12
Gen     ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                            65099
Next:                                                                       Auto.
                       A                        B           C
 1                                         1|   10|  0.1736481776669З|
 2                                        10|   20|  0.342020143325669|
 3                                        90|   30|                0.5|
 4                                         1|   40|  0.642787609686539|
 5                                         1|   50|  0.766044443118978|
 6<                                       90>   60|  0.866025403784439|
 7                                         0|   70|  0.939692620785909|
 8                                          |   80|  0.984807753012208|
 9                                          |   90|                  1|
10                                          |     |                   |
11                                          |     |                   |
12                                          |     |                   |
```

The SET command has one drawback compared to manually entering all of the values, because THE CRACKER is not always able to allocate order of recalculation numbers to the cells that it has SET itself. The next example will illustrate how this can cause some minor problems.

You are going to use the tables in columns B and C to interpolate a value for 25 degrees. To determine the value, enter the INTERP function in the following form:

```
A9                                                                          C12
Gen   Enter number or expression                                            65099
  >>  INTERP(25,B1...B9)                                                    Auto.
                       A                        B           C
 1                                         1|   10|  0.173648177666693|
 2                                        10|   20|  0.342020143325669|
 3                                        90|   30|                0.5|
 4                                         1|   40|  0.642787609686539|
 5                                         1|   50|  0.766044443118978|
 6                                        90|   60|  0.866025403784439|
 7                                         0|   70|  0.939692620785909|
 8                                          >   90|                  1|
 9<                                         |     |                   |
10                                          |     |                   |
11                                          |     |                   |
12                                          |     |                   |
```

An error message is returned saying that there is an ambiguity, this is because you are operating on a calculated table and THE CRACKER cannot resolve the correct calculation number to give the new entry, relative to the table.

```
A9                                                                    C12
Gen   Ambiguity (see manual)                                          65099
  >> INTERP(25,B1...B9)                                               Auto.
                          A                         B          C
  1                                           1|   10|   0.173648177666931
  2                                          10|   20|   0.342020143325669|
  3                                          90|   30|                 0.5|
  4                                           1|   40|   0.642787609686539|
  5                                           1|   50|   0.766044443118978|
  6                                          90|   60|   0.866025403784439|
  7                                           0|   70|   0.939692620785909|
  8                                            |   80|   0.984807753012208|
  9<                                          >    90|                   1|
 10                                           |     |                    |
 11                                           |     |                    |
 12                                           |     |                    |
```

You can solve this problem by first putting a reference to the cell that created the table. In this case it was the DO function cell, so you have to put A7 before your INTERP function. Putting this reference inti the formula informs THE CRACKER that the new function should have a calculation number greater than that of the DO function.

```
A9                                                                    C12
Gen                                                                   65099
  >> A7,INTERP(25,B1...B9)                                            Auto.
                          A                         B          C
  1                                           1|   10|   0.173648177666931
  2                                          10|   20|   0.342020143325669|
  3                                          90|   30|                 0.5|
  4                                           1|   40|   0.642787609686539|
  5                                           1|   50|   0.766044443118978|
  6                                          90|   60|   0.866025403784439|
  7                                           0|   70|   0.939692620785909|
  8                                            |   80|   0.984807753012208|
  9<                                          >    90|                   1|
 10                                           |     |                    |
 11                                           |     |                    |
 12                                           |     |                    |
```


Note the DO function has a calculation number of 6 and the INTERP follows it with 7.

```
A9     (A7,INTERP(25,B1...B9))                                        C12
Gen     ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                      52481
Next:                                                                 Auto.
                          A                         B          C
  1   <1>  (INIT(A2,1))                        |     |     (set)      |
  2        (set)                               |     |     (set)      |
  3   <2>  (10*A2)                             |     |     (set)      |
  4   <4>  (SIN(A3))                           |     |     (set)      |
  5   <5>  (SET(CRD(3,A2),A4))                 |     |     (set)      |
  6   <3>  (SET(CRD(2,A2),A3))                 |     |     (set)      |
  7   <6>  (DO(A3...A5),INC(A2),WHILE(A2<10))  |     |     (set)      |
  8                                            |     |     (set)      |
  9<  <7>  (A7,INTERP(25,B1...B9))             >     |     (set)      |
 10                                            |     |                |
 11                                            |     |                |
 12                                            |     |                |
```

If you use the X command you can switch the display back to see the result of the interpolation which gives value exactly half way between the sine of 20 and the sine of 30 as you would expect.

```
A9    (A7,INTERP(25,B1...B9))                                         C12
Gen     ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                      65067
Next:                                                                 Auto.
                          A                         B          C
  1                                           1|   10|   0.173648177666931
  2                                          10|   20|   0.342020143325669|
  3                                          90|   30|                 0.5|
  4                                           1|   40|   0.642787609686539|
  5                                           1|   50|   0.766044443118978|
  6                                          90|   60|   0.866025403784439|
  7                                           0|   70|   0.939692620785909|
  8                                            |   80|   0.984807753012208|
  9<                        0.421010071662834>     90|                   1|
 10                                           |     |                    |
 11                                           |     |                    |
 12                                           |     |                    |
```

The CRD function should only be used to specify a taget cell for the INIT, SET, INC and DEC functions. It is only a pointer and so cannot be substituted in every instance where you would normally specify a cell coordinate directly. For example, it cannot be used to determine the value of a cell in other expressions such as 2+CRD(1,2) instead of 2+A2. There is another function that will do this, the VAL or value function which returns the current value of the cell to which it refers. The arguments for this function are formed in the same way as for the CRD function.

Here is an example of the correct use of the VAL function, together with an incorrect use of the CRD function.

```
A1                                                                  B3
Gen    ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                     65457
Next:                                                               Auto.
              A                          B
  1<                        >                  3|
  2                  CRD2,1)=|                 0|
  3                 VAL(2,1)=|                 3|
```


ITERATIVE SOLUTIONS - USING CIRCULAR REFERENCES

As mentioned in the opening sections of this manual most spreadsheets will get very confused if you try to enter formulae that refer to each other in a circular way - e.g. making A1 equal to 2*B1 and B1 equal to 3*A1. THE CRACKER, however, is more broadminded about such things. Not only will it let you set up such circular references, but, if correctly implemented, they can be a powerful problem solving tool.

By setting up circular references to problems that are resolvable, you will see that THE CRACKER makes an estimate of the cell's values. If you then force repeated recalculations, this estimate moves closer and closer to the correct solution until an answer is found. This is known as an iterative method of problem solving. Such repeated calculations can profitably be automated using a DO WHILE function.

Consider the next practical example.

SOLVING SIMULTANEOUS EQUATIONS

Simultaneous equations can be easily solved using THE CRACKER. The technique used applies also to awkward equations where the unknown variable appears on both sides of the equation. The principle is quite simple, you prepare a set of formulae for the equations as if all the variables have known values, of course they don't because each formula depends on the results from the others and hence this is a circular reference.

When the last formula has been entered you force a recalculation and each formula uses the latest results availableg, thereby improving the result that each calculates. Further recalculations then bring the results nearer to the true results. This is a powerful technique that is often used for programmed solutions to equations.

Here is a simple example to demonstrate the principles, that has two equations and two unknowns.

The equations are:

$5x + 3y = 19$

$2x + 7y = 25$

Enter them onto the sheet using the constant multipliers of x and y, and ignore the values of x and y themselves, i.e. in the form:

```
C2    (25)                                                        C5
Gen   ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                    65445
Next:                                                             Auto.
          A                   B                   C
  1                5|                  3|                 19|
  2                2|                  7<                 25>
  3                 |                   |                   |
  4                 |                   |                   |
  5                 |                   |                   |
```

Now you enter the formula for the solution - assume that the final solution for x will be in A4 and the final solution for y will be in B4. You can, therefore, refer to these cells as if they hold the answers you are seeking. Use the first calculation to get x in terms of y. This is equivalent to manually rearranging the first equation with x on the left and everything else on the right.

x = (19 - 3y)/5 Insert this in A4

```
A4                                                                C5
Gen   Enter number or expression                                  65445
   >> (C1-B1*B4)/A1                                               Auto.
          A                   B                   C
  1                5|                  3|                 19|
  2                2|                  7|                 25|
  3                 |                   |                   |
  4<                >??????????????????????|                   |
  5                 |                   |                   |
```

B4 (or y) is unknown, but refer to it nonetheless. Next rearrange the second line in terms of the y unknown.

y = (25 - 2x)/7 Insert this in B4

```
B4                                                                C5
Gen   Enter number or expression                                  65421
   >> (C2-A2*A4)/B2                                               Auto.
          A                   B                   C
  1                5|                  3|                 19|
  2                2|                  7|                 25|
  3                 |                   |                   |
  4              3.8<                  >                   |
  5                 |                   |                   |
```


Note that A4 (or x) is used in this equation even though it doesn't have a true valueg - in fact, it has a first estimate of 3.8 - so these two equations rely on the answer from each other.

```
B4    ((C2-A2*A4)/B2)                                             C5
3Dec  ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                    65397
Next:                                                             Auto.
          A                   B                   C
  1                5|                  3|                 19|
  2                2|                  7|                 25|
  3                 |                   |                   |
  4            2.000<              3.000>                  |
  5                 |                   |                   |
```

Finally, you only have to press the ! key to force a recalculation a few times and the answers will settle down to the true answers of 2 and 3.

```
B4    ((C2-A2*A4)/B2)                                             C5
3Dec  ABCDEFGIJLMNOPQRSTUVWXZ!+-\/>.('* arrows                    65397
Next:                                                             Auto.
          A                   B                   C
  1                5|                  3|                 19|
  2                2|                  7|                 25|
  3                 |                   |                   |
  4            2.309<              2.486>                  |
  5                 |                   |                   |
```

The answer is reached after four recalculations, although more complicated equations will obviously require more recalculations.

THE CRACKER
OVERVIEW AND RECAP OF TUTORIAL

This section can be read by all new users of THE CRACKER who have some previous experience of spreadsheets, as a substitute for the tutorial guide. It is designed to give you a quick grounding in the way THE CRACKER operates and to ensure that you understand the philosophy behind the use of the various sections of the program. It is not intended to cover all the features of the program but you should be able to obtain pointers to where you will find more detailed information.

Necessarily, certain points will be covered that are common to many spreadsheets, however, you will find some features that go far beyond the abilities of any other program. It is well worth persevering, to be sure that you appreciate all of the program's features.

THE CRACKER'S DYNAMIC DATA ENTRY AND ERROR CHECKING FFATURES

THE CRACKER has a unique system of dynamic prompting and error checking, that provides you with full details, at any stage, of every possible command you can enter . Commands are input as one-letter mnemonics but, if space permits, a full explanation of all the permissible entries is shown on the prompt line as you type. More extensive help can be obtained by pressing the ? key.

As you enter a formula, THE CRACKER evaluates each command as it is completed. Any mistakes in the entry are immediately pointed out.

A cell can be referenced even though it does not contain any data, in which case it will return a zero value and be filled with question marks on the sheet. These question marks signify missing data.

The automatic recalculation feature can be switched off if desired, and indeed this may be advisable if you are entering any long and time consuming loops in a formula.

In normal circumstances both the prompts and the constant calculation are useful safeguards against error; there can be no ambiguity over exactly where a mistake has occurred.

See the section **More Details on Error Messages** if you are still unsure what has gone wrong.

At all times the [DEL] key (or backspace on some machines) will take you back to the previous step in the entry process. Mistakes are thus corrected. Pressing [ESC] will take you back to the primary command options available, except when you are using the EDIT option to make changes either to cell data, or a formula that already exists. In this case, no prompting or error checking is performed and it is often safer and easier to simply re-enter new data into the cell.

See the Edit Command section for full details on the available options.

THE SPREADSHEET AND ITS USES.

THE CRACKER is designed to use the available memory space of your computer to the full. The size of the sheet can be anything up to a maximum of 52 cells, labelled A-Z and a-z, and 255 rows, and no matter what size is defined, no memory is used until you start to enter data.

However, the number of cells that can contain data is limited by the amount of free memory your particular computer system and DOS provides. It is not possible to accurately predict this number, because it is dependent on the data entered into each cell. You can keep track of the remaining memory using the figure displayed at the top right of the screen.

If the memory becomes full, THE CRACKER will not allow any of your work to be lost, therefore it is as well to plan in advance whether a large project will have to be split over two sheets.

Before you can enter any data into the spreadsheet, you have to first define the depth and width of the sheet. The size of the sheet is set by using the Insert command to add the required numbers of Columns and Lines. The display width of each column is also set at this stage, although it can be changed again at any time.

You are also asked to specify a default display format for each column that you enter. There is a defineable, global format that will be used if you wish to default this choice. Individual cells can be assigned their own formats regardless of the column defaults.

As well as the number and size of both lines and columns, you are also prompted to specify WHERE you want these inserted. The defined sheet always has a rectangular shape. The line and column labels will be adjusted to reflect the new layout and so will any cell formulae that refer to specified coordinates.

See the Sections on Destinations and Adjusting References for more details.

Each cell is only one character high, but the columns are of any width that you care to choose, up to a maximum size of 24 characters. The column boundaries are shown on the screen by an upright line of vertical bar characters i.e. '|'.

Lines and columns can be removed from the sheet by the use of the Zap command. If you use this the size of the defined sheet will actually reduce. Line and column labels and cell formulae will again adjust to the new layout. Data that is erased in this way is not recoverable. An alternative approach is to use the Blank command which removes the contents of a cell but leaves the cell position itself in a defined state. Again the data is lost.

You are not allowed to Zap cell coordinates or blocks that would create a 'hole' in the sheet, nor can you remove data in cells that are used by formulae in another cell. However you are allowed to use BLANK in any of these situations and in the case of the deleted data you will see some question marks to remind you that some data is expected.

MOVING AROUND

You can use cursor keys, if you have them to move around the sheet you have defined. As an alternative use the L,R,U,D keys for left right etc. or the diamond shaped key cluster CTRL + WADZ.

The current cursor location is signaled by the presence of two angled brackets on either side i.e. < >, it may differ on your terminal. To enter any data to the current cell we must use the dot command '.' to go into Entry mode.

Any data, text or numeric, that will be used by the program in its calculations or other manipulations must by necessity be less than 127 characters long, but unless you are loading in a file created from another source the actual limitation will be the width of the entry line on your screen. Note that the data is not restricted by the apparent size of the cell on the screen, which can be shortened down to only one character wide. THE CRACKER will display as much of the information held in that cell as is possible given the limitation of the display and of the display format. The true value of the data is used in all calculations regardless of the way it looks in the display.

As an exception to the above display rules text can be entered in Heading format which means it will all always be visible. If necessary it will spread other several cells to act as comments on the data that is being displayed.

Instead of entering data directly special inbuilt functions CRD and SET can be used to create and fill cells on the basis of a calculation or formula. This is a useful feature that can be used to produce tables automatically. See the Section on Table Filling.

The third way of creating and filling cells is by use of the COPY function for copying a specified portion of the sheet into a new location. If you choose this option you will be prompted to decide whether or not you want the formulae held in the Cells that have been moved to be changed to reflect their new position.

TYPES OF ENTRY

Cell entries fall into two broad format categories, text and numeric. Each of these can have several sub-categories of display which can be freely interchanged. Text and numeric classifications cannot be interchanged since the data can only belong to one or the other. Text data can of course freely contain numbers and formulae, for example as an explanatory comment for the accompanying data. Even so it is impossible to change the data to a numeric format or calculate with it.

If a numeric format is chosen each cell can be assigned a numerical constant as an entry, for example 3, 9000, 7.88401, or it can be given a formula value which calculates the number that should be displayed on the basis of the contents of the other cells in the sheet or of a calculation of constants.

Simple examples of valid formulae would be

3+B1, 10*(343-A11), B23-C12/D13

Unless you tell it to do otherwise, by switching off the automatic calculation feature, THE CRACKER will attempt to calculate the result of the formula that you type in as it is entered.

Blank values in cells referred to in a formula will be taken as zero.

Formulas can also include within them complex functions, either built into the program already or those that are defined by you. Examples would include

B1+SUM(A10...A20)	Add the contents of B1 to the sum of all the cells in the range A10 to A20
SIN(30)-COS(A23)	Work out the sine of 30 degrees and subtract from it the cosine of whatever value is held in A23.

In these examples SIN, COS and SUM are all built in functions that can be used freely in your calculations. Any cell that is referred to in a formula, such as A23 in the example immediately above, can itself contain a value that is dependent on the solution of a formula entry.

It is therefore usually important to avoid circular references such as setting cell B1 equal to 2*A1 and setting A1 to 3*B1 for example, a situation that is not resolvable unless both cells take a value of zero.

Some circular references are logically possible, for example set cell A1 equal to 2*B1 and set cell B1 to equal A1/2 and in certain special circumstances it is possible and indeed very useful to be able to enter a circular reference of some kind for the iterative solving of simultaneous equations. Consult the Section of that name in the

Tutorial Guide.

A full list of built in functions available is given later in the Expression Entry section of the Complete command summary. The expressions and formulae you build up using these functions are the processing heart of THE CRACKER. It is these that make it such a useful tool. They can handle simple mathematics up to the most complex financial or scientific calculations.

£ and ! are two useful commands that come into use when entering data. If £ is typed after a cell coordinate when entering an expression then the reference is replaced by the actual value that is held in the cell at that time. If the value that is held in the cell is later changed it does not affect the expression that has been entered.

The ! command is similar, but it causes the entire expression up to the point where it is typed to be replaced by its calculated value.

It is possible to switch between the normal spreadsheet display of numbers and text and a display of the formulas that go to make up these numbers by use of the eXchange command.

MORE ON FORMATS

At the time you are entering inserting new columns or data or you will be prompted to state which FORMAT type should be used for that particular cell, i.e. how you want the data to be displayed.

To save time it is possible to assign a default Global format for the whole sheet or any line or column within it. Individual cells can still be assigned format types of their own.

On first loading THE CRACKER the entire sheet is given a default left justified text format and a General numeric format which is like a scientific calculator.

See the Format Command section for more details.

There are about a dozen built in formats that can be used, some relating to text and some relating to numerical data. It is possible to switch formats as long as the data within a given cell can conform to the new type. For example we can switch a number between Integer format, which only displays the whole number part of a value, to a Financial format, which displays data correct to two decimal places.

Data can be assigned to an incorrectly formatted cell if it has been loaded in from a disk file onto an existing sheet or by using the edit option. However any attempt at performing a calculation on the data will throw up the error.

As is the case when changing the display width of a cell, changing a format will change the way that information is displayed on screen but will not alter in any way the actual value of the information itself. For example if you have altered a cell value of 2.5345 this value will be used in any calculation regardless of the type of format used for the display. This can result in apparent calculation errors, for example in integer format two times the above value would appear to give the result that 2*2 = 5. A true integer calculation can be forced by using expression such as INT( ) in the calculation formula.

Certain of THE CRACKER's inbuilt functions and expressions work on a specified range or column of the data. For example to AVERAGE some of the data you would specify the range thus

AVERAGE (B1...B10)

Any cells that conform to an incorrect format, for example TEXT, that fall within this range, or any blank cells, are ignored in the above calculation. This saves you from having to create unnecessarily complicated expressions in order to encompass all of the data required.

Careful use of the Format option can help to produce some quite sophisticated displays and printouts. Text can be left or right justified in its Cell. There is also a default format choice of Carriage Return that can be assigned to individual Cells or ranges.

For example these can be used to control printout in such a way that address labels can be produced - see pages 99-103.

THE COPY COMMAND

One of the most important commands available is COPY. It's use extends far beyond that of simply copying data from one part of the sheet to another, it is designed to allow any information to be written from any input device to any part of the sheet, or from any part of the sheet to any output device, or between any two parts of the sheet. It therefore takes the place of both a save and load command and of the printing command. As well as saving, loading or printing entire files it is also possible to perform operations on defined blocks or ranges of the data. 'Blocks' as small as one Cell can be copied to the disk or printer. Once the option has been selected you will be prompted through all of the available choices.

When copying within the sheet you cannot copy beyond the current sheet boundaries. Any data at the destination will be overwritten unless another part of the sheet refers to it. The moved data takes its own format details with it.

There are four possible file types that can be saved from, and conversely loaded into, THE CRACKER. These are specified by the type of filename extension given when saving or loading. The four types are

.MEM - THE CRACKER's own internal filing format
.DIF - the file is suitable for interchanging between most other spreadsheet or graphics programs.
.DAT - only numeric data is held in these files in a formatted ASCII form suitable for reading by text editors, FORTRAN or BASIC
.TXT - the file is saved in an ASCII text form suitable for reading into word processors etc. Text and numbers are saved.

More details of these filetypes are given below

The command sequence to load a file is CFname i.e. Copy File 'name' (to the sheet).

To save a file the sequence is CAFname i.e. Copy All of the sheet to the File 'name'.

Of course you can save sections of the sheet such as a block by specifying alternatives to the All command e.g CBname.

The move command is related to Copy, but is only applicable to movement of data within the worksheet.

FILE SAVING AND LOADING

THE CRACKER recognizes four types of Data File denoted by filename extensions .MEM .DAT .TXT and .DIF.

The standard file type that THE CRACKER uses for saving and loading of information is the .MEM file. Files saved with this extension will make a record of all the information held in the current sheet (or block within the sheet if specified) and of the way in which this information should be displayed (the Formats). It therefore follows that any file that is loaded in with the .MEM extension will be expected to consist of the correct type of data.

The filename extension .DAT will save only those parts of the sheet that contain numeric data.

The filename extension .TXT will save all numeric and also text data.

Both filetypes save the data in an ASCII form, both can be read into word processors or other text editors.

You can also use this option to load in worksheets that have been created on text editors or word-processors such as Newword. It is not necessary to ensure that the file has been saved as a simple ASCII character file, THE CRACKER is able to read and remove the Control codes used by many word processors.

At any time the key combination CTRL-F will save a complete text copy of the part of the screen that is visible at the time with the name DUMPn.SCN.

The filename extension .DIF will save data in what is known as a Data Interchange Format which is a standard system used for sending information between different spreadsheets or from spreadsheets to graphics packages. It is not necessary for you to be able to understand the precise details of this format, but it is possible that you will have to make some adjustments to the data display when it has been loaded in.

Files with the extension .BAC are also displayed whenever a disk directory is requested and it is possible to rename these so that they can be loaded into the program.

DAT and TXT files save the data as it appears on the screen based on the current format, including truncation if the column width is too small.

Word processors and programming languages can be used for creating the latter two files.

When loading files in THE CRACKER anticipates the type of data it will find depending on the extension given. When loading TXT or DIF files it is necessary to define the size of the file first.

If you wish to insert a file into a sheet that already exists it is necessary to create room for the incoming file. Alternatively the incoming file can be simply appended to the existing one.

Blocks that are saved or loaded (or blanked) must be self contained, the formulae within it must make no references to cells that are outside of the block.

It may be necessary to ensure that the defined formats of an existing sheet are of the correct type for the incoming file.

If you use the Quit option to leave THE CRACKER without having previously saved your work then THE CRACKER automatically makes a copy of the sheet called SECURITY.MEM. This file should be renamed or resaved as soon as possible so that it is not accidentally overwritten.

PRINTING

Worksheets, or a portion of them, can be printed by using the Copy to Printer command sequence. Sheets are printed as they appear in the display but without dividers or system messages.

More information is found in the section on printing in the Tutorial Guide.

Alternatively pressing the CTRL-P keys will print a 'snapshot' of the currently displayed screen.

Mail labels are a special print option but to use them to their full advantage will involve inclusion of special columns containing Carriage Return default formats. Again see the appropriate tutorial section.

DATABASE FUNCTIONS

The worksheet can be searched to find a specific data entry, either text or numeric. The search will operate on the true data value or expression that has been entered in the cell NOT the displayed data. See the GET command.

Columns of data can be sorted into order, i.e. physically re-arranged in the sheet such that the information in the specified column ascends or descends.

It is best to sort either text or numeric data, unpredictable results occur if they are mixed. If the data is text then upper case letters are treated the same as lower case, numbers rank below letters and are treated in a textual way e.g. 7 ranks higher than 66. See the Sort command.

ADVANCED FUNCTIONS

Conditionals, Loops and Macros

THE CRACKER includes some features that will be familiar to anyone who has had experience of high level computer languages such as BASIC.

The conditional command sequence IF (logical test) THEN XXX ELSE YYY can be used to build decision making into the sheet. If the logical test is passed as TRUE the part of the expression after the THEN is calculated. If the test is FALSE then the ELSE part is calculated. page xxx

An extension of the IF THEN sequence is the DO XXX WHILE (logical test) loop. The expression following the DO is performed at least once, and then repeated over and again whilst the logical test is passed as TRUE. As soon as the test becomes FALSE or an error occurs then the loop is stopped. The simplest form of logical test is to set up a counter that is increased or decreased with every DO expression, special functions INIT, INC and DEC help you to do this; when the counter reaches a specified value the loop will stop. There is a lot to learn about these commands and you are advised to read pages 151-154.

A MACRO is a term given to a sequence of commands that you are able to access with just one command. They are most useful when you find that you are performing a certain sequence of commands over and over again. Looping Macros can be defined, i.e. ones which continue to repeat until the task is completed or an error occurs.

Note that a Macro is a loop of direct commands rather than of mathematical functions such as occurs with a DO WHILE loop. It can contain commands such as L for cursor left, or NF for new Format, commands that act directly on the sheet and can not be made part of an expression. The Macro commands act exactly as if you have typed them in at the keyboard. Page 245.

TABLE CREATING AND READING

Certain functions, (SET, INIT, DEC, INC) exist that will let you set the value of data held in another cell. These can be incorporated into a DO WHILE loop in order to facilitate the automatic creation and filling of tables.

Other functions exist such as LOOKUP which are designed to automate the process or consulting tables in order to extract data.

DATE AND TIME FUNCTIONS

THE CRACKER will let you input data and time functions into the sheet. The program can be given details of the present data and time and, as long as the program is running, it will keep a track of these. It is then possible to build these functions into formula expressions, logical tests,loops and macros such that the program will adjust its output dependent on time. A full list is given on pages 214-215.

GRAPHICS OPTIONS

A range of graph and chart designs can be created through THE CRACKER's Trace Graph option. Special functions exist to signal to THE CRACKER which type of graph you want to display and to inform the program of which data ranges are to be plotted together with labels and axis scales etc. See the Trace Graph command.

The options are extremely simple to use but require the use of some disk overlay files. This may mean that you will have to give more thought to which files you put on to which discs. See the Section on Getting Started.

THE CRACKER QUIKSTART GUIDE

INTRODUCTION

It's impossible in any manual to be able to arrange the information to suit everybody. It is more important to present the information in a logical order and make it easy to find.

For those of you who are to impatient to take each step in turn or who have to get on with an important job here is THE CRACKER Quickstart guide to fundamental tasks.

STARTING A NEW SHEET

Step 1

Begin by defining your worksheet.

Press Command I to Insert between 1 to 52 Columns. Specify the display width of the columns, 1 to 99 characters. New ones can be added at any time and the width of existing columns can be changed. Pages 26-28.

Specify the default data display format for the column(s). The most important choice is between text and numeric formats - text is specified as TL (left justified), TR (right justified) or Heading (always displayed in full despite the column width).

Numeric formats are every other choice except Carriage Returns.

Formats do not actually alter the data you enter, just the way it is displayed.

A full explanation of Formats is given in the Tutorial guide.

Step 2

Press command I again to Insert between 1 and 255 lines.

Step 3

Use the cursor keys or equivalent to move around the sheet.

Step 4

Press the '.' key to be able to insert data in any cell according to the specified format type. If you are entering a lot of data do not press the Return key when finished but use the arrow keys to move to the next cell and you will remain in Entry mode.

Individual cell formats can be specified using the F for Format command.

A full list of the functions, mathematical and otherwise, that can be included in a numerical data entry is given in the Expression Entry Section in the Complete Command Summary.

Keep the automatic calculation feature on and the data should always be displayed in an up to date form.

LOADING AN EXISTING SHEET

Step 1

Use the Which Files command to find out the names of the data files on the disk. If necessary change the logged drive and the user number or sub-directory.

Step 2

If you are loading a MEM file use the commands Copy File filename to load in the file.

If you are loading a file with the extension .DAT, .DIF or .TXT you should first define an empty sheet big enough to hold the data . You may have to load the data in once in order to determine the best layout and the best way to set the formats for the individual cells. Numeric data that is loaded into a text cell or vice versa will have to be re-loaded or re-written.

Only those four filename types are recognized. THE CRACKER expects each filename type to hold a certain type of data - do not use them indiscriminately. When in doubt use the MEM extension.

See the Tutorial guide for more details

A file can be added on to one already in memory, if there is room, using the CF command. You will be asked to specify where the file is to go.

SAVING A SHEET

Step 1

Decide whether you want to save the whole file or just part of it. If you want to save a part specify the cell, line, row, or block that you want to put onto disk.

Step 2

Use the Copy All (or named part) to File 'filename' command sequence to save the sheet.

Use the filename extension .MEM (or no extension) for re-loading into THE CRACKER. Use the .DIF extension for loading into another spreadsheet or graphics program.

Use the TXT or DAT extensions for loading into a word processor or for processing by BASIC or FORTRAN language programs.

Step 3

If you get the message Disk Full the insert a new disk and try again. It is advisable that the new disk is blank.

PRINTING

Step 1

Decide what area of the sheet, if not all, that you want to print.

Step 2

If the width of lines is wider than your printer can manage print the sheet in two or more sections.

Alternatively use the OUT command to select condensed print (if your printer is capable of it). See the Section on this command in the Complete Command Reference.

Step 3

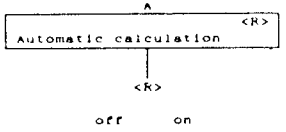
Use the Copy All (or specified section) to Printer command to print the data.

Data is printed as it appears on screen, without line dividers or system messages.

Study the section on the CTRL-P command on page 180. Also read the Section on Printing Mail Labels.

THE CRACKER COMPLETE COMMAND REFERENCE

AUTOMATIC CALCULATION COMMAND



Automatic calculation, or perhaps it should be called automatic recalculation, is the feature by which as you enter any new data or formulae, or replace existing data by something new, THE CRACKER automatically updates the displayed worksheet in order to fully reflect these changes.

You can turn on the automatic calculation by typing an A followed by a RETURN. Similarly a second entry of A followed by RETURN will turn it off again.

As you get more experienced with the program you will probably find it useful to switch off the calculation process while entering long or complex formulae or perhaps large tables of data. The few seconds taken up each time to calculate and display the results can slow down the overall entry process.

When you switch on the automatic calculation again a full recalculation will immediately be carried out. If you wish to leave the automatic calculation off permanently then you can use the '!' COMMAND to force a recalculation at any time you would like it.

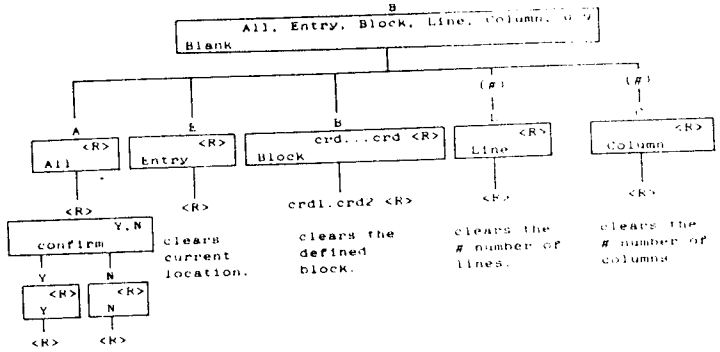
Automatic calculation proceeds AS YOU TYPE when you are entering new data or formulae. This is an important part of THE CRACKER's error catching features. As soon as a calculation or command has been entered that THE CRACKER cannot deal with the calculation will stop and you will know exactly where the mistake lies.

The only exception to the above rule is when you are using the EDIT command to make small changes to existing entries. In this case calculation must be suppressed until the editing session is completed or you will find that as soon as you delete a certain part of a

formula it is probable that an error will be found. THE CRACKER therefore automatically turns off the calculation until the editing is finished, at which time a full update of the sheet is made.

New users of the sheet may find it preferable to avoid using the EDIT function and simply delete and re-enter any formulae that you wish to change. You are then assured of the benefits of error checking and the command line prompts.

BLANK COMMAND



BLANK removes the specified entries from memory but does not affect the structure of the worksheet. Blanked entries cannot be retrieved so if in doubt you should first copy the unaltered worksheet to a .MEM file before starting on complex rearrangements.

Before the blanking is carried out the program will check whether any of the items to be blanked are referred to elsewhere in the worksheet. If cross-references are found the command will not be carried out.This is normally useful to prevent you from making mistakes. If you still want to erase the cells in question you must work backwards through the logic of the sheet removing all of those formulae that depend on a particular entry value, before you can remove the value itself. In practice you will find that this is simply done by just working back through the 'order of calculation' numbers.

If you are deleting an item referred to by some function such as SUM(A1...A5) which work on a range then the blank will be accepted but you may be asked for confirmation before it is carried out. Range functions will normally just ignore any blank or text entry that falls within the specified list.

CONTROL KEY COMMANDS

CTRL-C Abort command, returns directly to CP/M.
CTRL-S Pause command, operations will cease until CTRL-S is typed again.
CTRL-F Snapshot taken of screen and put to file.
CTRL-P Snapshot taken of screen and printed.

These are four special commands all initiated by holding down the CTRL key and at the same time typing the character following the dash.

CTRL-C should only ever be used with extreme care as no copy of your work will be saved. As a general rule always use the QUIT COMMAND instead.

CTRL-S is a pause command that will suspend operations until a second CTRL-S is typed.

CTRL-F stops operations immediately and takes a copy of the screen, including status lines and dividers, and sends it to a file. The first file will be named DUMP-A.SCR and if you then take a second snapshot the file will be named DUMP-B.SCR and so on. These files are standard text files that can be handled with an editor or word processor. They can only be read by THE CRACKER if they are first renamed with a .TXT extension. On completion everything carries on as before. This command causes only a temporary interruption.

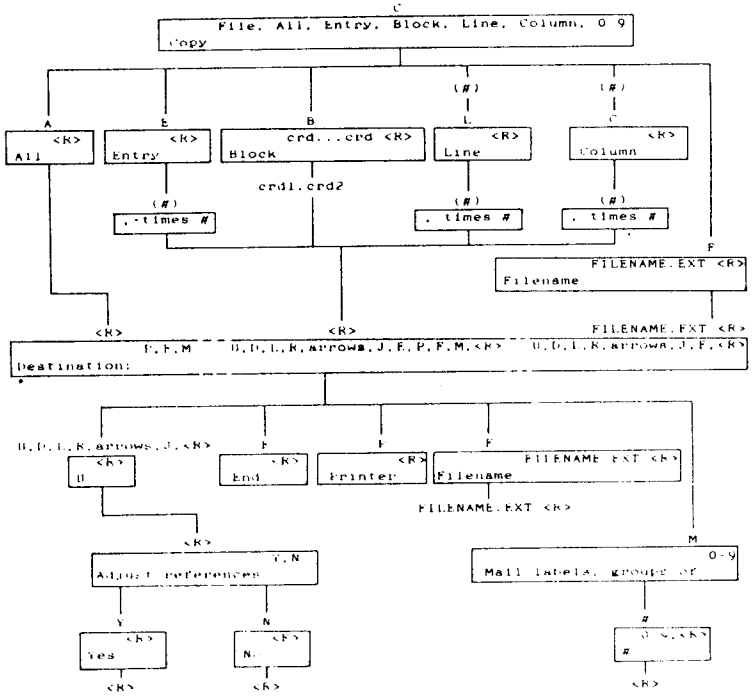
The files so produced are ideal for incorporating portions of the worksheet into reports produced on word processors etc. The filetype so produced is in many ways the same as that which can be produced by copying a defined block to the disk using the .TXT filename extension. Although the latter is more work to set up it does have the advantage that it can cover a larger part of the sheet than can be displayed on the screen at once and it is free of possibly unwanted status messages.

CTRL-P stops operations and immediately sends a copy of the screen to your printer, complete with dividers and status lines.

Again a data - only printout can be produced by using the Copy command on a defined block.

If there is not enough spare memory then the CTRL-F and CTRL-P facilities will be switched off to give you more workspace. In this case you will just hear a beep if you try to use them. No harm will be done.

COPY COMMAND



The COPY COMMAND is used to transfer copies of sections of the memory, displayed results or entire files to other locations, other files or the printer. There are restrictions with just what can be transferred to where but even so this is a most useful command.

Copying within the worksheet

You have three options. Firstly you can make a single copy of an entry, line, or column to somewhere else in the sheet. Secondly you can copy several lines or columns at once to a new area. Thirdly you can copy one entry, line or column several times.

Generally when making a copy you will be asked if you want to adjust the references. This means that if you are copying a line, all references to other locations on that line will be changed to the destination line. This preserves the sense of the calculations along a line. The same applies to adjusting references down a column and by extension in a block.

Copying to a printer

Copy also performs the role of a printer command and can be used to produce hard copy of specified data from the sheet, ranging from a single entry, lines, columns, blocks and the entire sheet. The printout of course reproduces the data as shown in the current display format and using current column widths etc.

Copying formulae to the printer

It is often useful to be able to print out a copy of the rules or formulae that underlie the worksheet display. To do this we must make the formula 'visible' by using the eXchange command. Next adjust the column widths so that they can be seen fully and finally copy the area in which you are interested to the printer.

Areas larger than the displayed screen can of course be printed.

When printing out large worksheets you may often wish to fit as much information as possible on each sheets of paper. It is possible to switch to condensed print, if your printer is capable of it, by using the OUT command to send the appropriate codes.

Preparing mail labels

This command will prepare printed mail labels. The addresses will usually be on a single line in the worksheet so you will have to insert markers where you require each next line to begin. This is done by inserting an extra column at the end of each address line which is given a default format of CARRIAGE RETURN. Remember to put in a carriage return column at the end of the address. You will get confused results if you leave it out. You will need to adjust the column widths to correctly align the addresses onto the tops of the labels. A more detailed explanation is given in the tutorial guide.

Copying from files into the sheet

You can copy a file into the worksheet from the current disk unit, or another if specified using the appropriate prefix e.g. B:FILENAME.EXT. The file may be put into any specified blank area of the worksheet or be placed as an addition at the end of the sheet. This command allows very flexible file merging.

The file to be read may either be in the format of a .MEM file or be a the file is to be copied is vacant and is large enough to hold the data and if not the command will not be implemented. Each file will be first read to determine its structure and then read for a second time to extract the data itself.

Each filename extension signifies to THE CRACKER that it should expect a certain type of information, and held in a certain form. The rules are as follows.

If you wish to get graphics from graphics packages or other spreadsheets use the use the .DIF format. You may need to rename the file to have the .DIF extension because THE CRACKER will not otherwise recognize it even though it has the correct format. Numbers will come in formatted to be General and text to be Text Left Justified.

To bring in data from a word processor or screen editor you must give the file a .TXT extension. THE CRACKER can read both normal ASCII text files and most DOCUMENT files created by word processors. These are converted to normal text files as they are read in. (Document files differ from standard text in that they contain certain, normally invisible, control codes that are used by the word processor to ensure that the right hand edge lines up etc.)

The data text should be laid out in tabular form. It is possible to bring in the numbers as values that THE CRACKER can use in calculations. The requirements are that you set up the column widths so that each column in the worksheet coincides with one column on the incoming file. The way to visualize the operation is to picture the text coming in as being laid directly on the top of the worksheet as it currently is. Wherever it lands THE CRACKER will try to interpret it sensibly. Remember that the column divider will count as a space.

You must also set the default formats for the columns to be suitable for the incoming file. Be careful that stray items of text in a number column cannot be sensibly calculated and so will stop the operation (without harm). To get round this problem try starting with all the default formats set to TEXT LEFT and check that everything looks suitable. If necessary delete items that would cause a problem. Then copy the screen back out to another .TXT file. (of course if there are no problem cells then there is no need to create a second .TXT file, you can just reload the first one). BLANK ALL and set the default formats to the final form and read in the second .TXT file. It's easier than it will seem on reading this.

Files with the extension of .DAT are expected to contain just numbers. The files can be created with a word processor or other editor (including document files) or as the result of formatted output from a program written in BASIC or FORTRAN.

Files with the extension of .MEM are intended purely for use from within THE CRACKER itself. They contain sufficient information for the program to be able to exactly reproduce the layout and display of the saved file.

Copying to files

You can copy to files on disk either in ways that are intended primarily for use with other programs or in such a way that THE CRACKER itself can read it and reconstitute it exactly.

There are two types of text file that can be read by other programs. The first with the extension .DAT will only copy numbers as they appear on the screen to the file. The second with the extension .TXT will copy both numbers and also any text, titles and headings, as laid out on the screen. Both will copy any part of the worksheet. If you arrange your .DAT file correctly before saving you should be able to read it from a BASIC or FORTRAN program if you want to do further processing.

Status lines and column dividers are not reproduced.

To copy to a file for reading by other spreadsheets or graphics packages just give your file the extension .DIF for data interchange format. Do your copying in the normal way and the program will make whatever adjustments necessary.

If you are copying to a file that already exists then it will not be destroyed. Instead it will be renamed with the extension .BAC. This applies in all cases including the SECURITY file. If you have a disaster you can read these files after renaming them with a .MEM or other appropriate extension. Use the WHICH FILES command to do this renaming.

However any existing .BAC files are erased, so with SECURITY files in particular it is important to remember to rename them as soon as possible.

CURSOR MOVEMENT

U, up arrow,	CTRL-W	upward movement
D, down arrow,	CTRL-Z	downward movement
L, left arrow,	CTRL-A	left movement
R, right arrow,	CTRL-D	right movement
+		cursor moves down one page
-		cursor moves up one page

A page is defined as the maximum number of worksheet lines that can be displayed on the screen.

Moving directly from one entry to the next

If you use the arrows, or the control-key cluster, while in the process of entering text or expressions then it will finish off that entry and move the cursor to the cell in the direction you have specified. This cell will be set up ready for you to make your next entry. The cell format will be set up to be exactly the same as the previous entry.

The use of arrows in this way can save much time with long lists as the one key stroke is equivalent to RETURN, cursor movement and '.' for entry. Obviously the LRUD keys cannot be used since THE CRACKER cannot tell whether or not you are intending to enter some text, a cell reference or a function name. See also the JUMP command.