INDEX

INSTRUCTIONS

File the index page from every issue at the back of your Fast Access binder for a handy reference to your library of programs. The pages can be ordered by Icon or by issue, it's your choice.

80 TRACK DRIVES

Insert FAST ACCESS into your disk drive and type:
*CHANGE <RETURN>
The program will prompt you to insert a formatted 80 track disk when necessary. Single drive users will have to swap the disk several times.

TITLE	PROGRAM FILES	TEXTFILES
Cassette Data Display	B.CASDATA	T.CASDATA
	O.CASDATA	
Cordelia	ADVRUNF	T.CORDEL
	ADVDAT	
	CORMESS	
	CORLOCS	
	I.CORMESS	
	I.CORLOCS	
	INIT	
Encrypter	ENY-SRC	T.ENCRYPT
	ENY	
Gallery	GALLERY	T.GALLERY
	GALSCR2	
	COMP	
Stylish Print	DEMO	T.STYLE
Gameplan Part II	GLOSS	T.GAMEPLN
	SCREEN	
GoldBank	GB-LOAD	T.GOLDBNK
	GB-CODE	
	GB-CHAR	
	GB-GAME	
Jumble	JUMBLE	T.JUMBLE
Knight's Move	KN-LOAD	T.KNIGHT
	KNIGHT1	
	KNIGHT2	
Sort Tutorial Part II	TEACH2	T.SORT2

ALL USERS

Make backup copies of both disks and keep the originals in a safe place with Write-Protect tabs on. Use only the copies, as many of the programs write to the disk, which will diminish the usefulness of the originals. For specific filing system information, please refer to the help file on disk.

NEW USERS

Don't Panic!. First find out whether you have 40 or 80 track drive(s) attached to your computer. Then go to your User guide or Welcome Manual and find out how to use the *COPY command. Next re-read the section above All Users, and then continue reading down from this point.

40 TRACK DRIVE SYSTEMS

FAST ACCESS is supplied on 40 track disks and will work on any 40 track BBC Micro system straight away. Remember to make a working copy before use.

80 TRACK DRIVES

If your filing system allows double-stepping, we recomend using the system's own command. As a general rule, built-in 40-to-80 track converters should be used where available; the documentation for your filing system or utility ROM will give full instructions.

ADRESSES

If for any reason your copy of Disk User will not work on your system then please carefully re-read the instructions given above. If you still experience problems then return it to: INFONET LTD, 5 River Park Estate, Billet Lane Berkhamsted, Herts HP4 1HL. TEL 0442 87661 to 4

Advertising enquiries to FAST ACCESS, Argus House, Boundary Way, Hemel Hempstead, HP2 7ST. TEL 0442 66551.

Editorial and technical enquiries to FAST ACCESS, 6C Belgic Square, Padholme Road, Peterborough PE1 IXF TEL 0733 53355.

Contributions should include full source code and instructions file on disk. Payments are extremely competitive.

Typesetting and artwork by Simon Fifield.

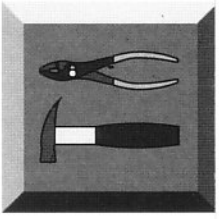
Printed by Loxleys, Sheffield.

Duplication by Direct Disk Supplies, Teddington, Middlesex.

RESOURCE FILES

TITLE	FILES	TYPE
Menu	FAMENU	BASIC
	GOSPR	Data File
	GOCODE	M/Code
Info Desk	IDESK	BASIC
Change	CHANGE	M/Code
Help	T.HELP	Text File
Ricochet Ad	POLISCR	BASIC
	POLIS	Screen File

CASSETTE DATA DISPLAY



AUTHOR

R.A.COWARD

FILES

'B.CASDATA'.....BASIC
'O.CASDATA'.....M/CODE
'T.CASDATA'.....TEXT

This program is an interrupt driven routine which displays all the useful Cassette and ROM filing system parameters in a clear and distinctive form at the top of the screen. The major application of this routine is for TAPE to DISC transfer, and in this area its usefulness cannot be over-emphasised. You no longer need to look at the last block of a file to determine the file parameters, and you can immediately find out whether it is locked or not.

For simple TAPE to DISC transfer of unprotected files, the top line of the display is the most useful, giving Filename, full load address, full execution address, locking status and flag byte. The latter is 80 or 81 hex for the last block in a file, and is 00 or 01 for intermediate blocks.

If the file is locked, it is 01 or 81 hex, and if unlocked, it is 00 or 80. This display is always available and clear to read. When you are dealing with protected files, however, the routine comes into its own; the second line is a full hexadecimal dump of the filename contents, to enable any invalid codes to be identified. Any spaces are shown as white blocks in the filename display, and invalid codes are displayed as '#' symbols, to differentiate them from the normal CFS display of '?', thus identifying true '?'s. In the filename hex dump, the display turns yellow after 00 has been detected (end

of filename). The flashing marker 'CDS' indicates that the routine is operating correctly, and will cease to flash should the routine be corrupted or interrupts are turned off.

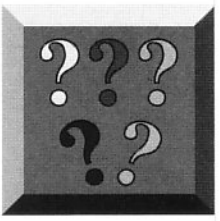
The bottom line of the display gives the full block number, full block length, bytes &3CB to &3CE (used in the ROM filing system to point to the start address of the next file), the filename being searched for by the controlling program and the value of PAGE. The latter two pieces of information are useful for checking out the operation of protected loader programs (eg. to see which file they are searching for and where they aim to store it in memory).

The object code is supplied as file O.CASDATA on the disk and can be directly *RUN. It currently occupies memory between &900 and &AFF, but can be assembled in a different place if desired. File B.CASDATA on the disk is the assembler program, and contains REM statements which clearly spell out the alterations required to relocate the program, as well as other modifications that can be made to suit personal preference.

Though the program will not display DFS parameters for obvious reasons, it will not interfere with DFS operation and it does not use any zero page workspace. The interrupt routine uses the vertical sync event, activated by *FX14,4 and deactivated by *FX13,4. In operation, the routine is transparent to the user, provided that the screen mode is not changed or reset (this will destroy the predefined text window and upset the screen layout). If this happens, simply re-run the routine. Also remember to switch off the routine before overwriting its workspace, or else the computer may crash.

This routine is not intended to be an aid to software piracy and has never been used as such in the past (a tape dubbing machine is all you need to pirate software). It is just a very useful routine to ease the problems of tape to disc transfer.

CORDELIA



AUTHOR

BASIL WATERTON

FILES

'ADVRUNF' ..... BASIC
'ADVDAT' ..... DATA
'CORMESS' ..... DATA
'CORLOCS' ..... DATA
'I.CORMESS' ..... DATA
'I.CORLOCS' ..... DATA
'INIT' ..... DATA
'T.CORDEL' ..... TEXT

Your daughter Cordelia has run away to Dominic whom you know to be a dangerous Devil worshipping Satanist. Cordelia doped you, took your car and removed the plug from your motor bike. Your parental instincts tell you that she is in grave danger and you must find her quickly.

THE SOLUTION

If you need help in your quest then read on but otherwise go to the menu on disk A and click on the Cordelia icon to run the adventure. The game takes quite a while to load as it is very BIG so please be patient.

To rescue Cordelia you must read a postcard which reveals where she will be, and fit a plug in the bike which you take from the garden mower.

You must find a map, wear a leather jacket and helmet and take with you a bottle of dope, some raw meat, a ball of string and scissors.

You ride the bike and arrive in Norton Aster where Dominic lives with his Satanist monks at Norton Towers.

You enter a barn and find a hook in the hay loft. You cut the ball of string into nine lengths and plait them to make a rope.

You tie the rope to the hook and gain access to the grounds of Norton Towers by climbing

the wall using your rope and hook.

In the grounds you are faced by savage dogs. You dope the meat and throw it to the dogs who eat and fall asleep. A gardener demands you give him the dope, thinking it to be milk.

You give the bottle of dope which he drinks and falls asleep. You take the empty bottle and enter the house. You take matches from a store room and then take and wear a mask and a cloak.

In the library you meet Dominic but if wearing the mask he fails to recognise you. You visit his bedroom(points gained) then return into a maze and find an altar and a dagger. Take the dagger and back through the maze.

You climb the tower and find Cordelia bound and unconscious. Cut the bonds but remove the mask first, or she comes round and kills you! Take Cordelia and leave the house.

Search the coal shed and take a hammer. Fill the empty bottle with paraffin. You then meet a Satanist monk. He will kill you if you try to pass. Throw dagger will kill him, and he drops a key. Take key.

Unlock west gate then open it. Go west, close gate and lock it. If you don't lock it you are caught. Go north. If you go south you are caught.

Search at the dead end, where you find graves of Dominic's sacrificial victims. You also find steps down to a passage. An oil lamp is on the top step. You take the lamp.

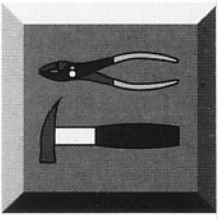
Fill the lamp with paraffin and light it, then go down. A passage leads to a cave where dead victims are dumped. The main passage now enters a maze. The roof falls in killing you in one location, so a SAVE is very advisable.

You arrive at a trap door which has a rusty bolt. Hammer the bolt to free it. Up brings you into the barn where you earlier found the hook.

You return to the village and enter the phone box. It was engaged earlier in the game. You dial 999, then say POLICE. This concludes the game.

The above is a very shortened description of the adventure. The Complete solution can be found on disk B.

ENCRYPTER



AUTHOR

GARY FIELD

FILES

'ENY-SRC' .....BASIC
'ENY' .....BASIC
'T.ENCRYPT' .....TEXT

Some months ago I was trying to think of a way of stopping prying eyes looking at my programs (or using them).

I came up with a machine code program which encodes a BASIC program to a password. The encoded program can still be saved and listed but it will look like rubbish and will therefore not run.

ON THE DISC

The program 'ENY-SRC' is the source code for the machine code encrypter. When run, it will save the machine code under the file name 'EN-CRYPT'.

The two line program 'ENY' is an example of what the first two lines of a program to be encrypted should look like.

The machine code program has been preset to miss out the first two lines of a program (when encrypting) so that a password can be entered after the encryption. The number of lines it misses out can be altered by changing the value of the variable 'lines' in the program 'ENY-SRC'.

HOW TO USE ENCRYPTER

Using the encrypter is simple. All you do is add the 'ENY' program (2 lines) to the program you want encrypted (after copying 'ENCRYPT' to the program disc) and run it. (You can use your own version of 'ENY').It will ask for a password, so you type one in (e.g. BREAK-IT or JACK) and hey presto! the program looks

like garbage when listed. Don't worry if the computer displays 'Syntax error' when you enter a password, it is because it can't run the rest of the program.

You can now save the program as a normal BASIC program.

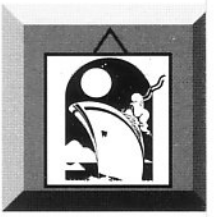
When the program is RUN or CHAINed you must enter the correct password and the program will be decoded and run. The good thing about this program is that the password is not stored in memory after the program is encrypted.

Because Encrypter uses EOR to encrypt programs, you can use as many passwords as you want (making the program even more secure).

As long as you decode the program with each of the passwords (i.e. repeat 'ENY' for how many passwords you have), you will get the program back.

It is virtually impossible for anyone to read the program without knowing the password (including yourself so be careful!).

GALLERY



ARTIST

'STEVEN NISBET' ..... FACE BEHIND THE MASK

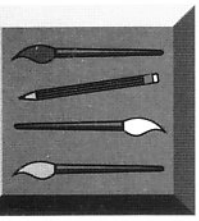
FILES

'GALLERY' ..... BASIC
'GALSCR2' ..... COMPRESSED
'COMP' ..... MACHINE CODE
'T.GALLERY' ..... TEXT

In this issue's Gallery we have another great painting by Steven 'Hawk' Nisbet. He paints using the Quest Mouse and his own custom art program. The quality of his artwork is second to none as you will see if you select the Gallery icon from the menu screen on disk B.

The file 'GALSCR2' on disk is a compressed screen file. To display the screen a program called 'comp' must be *RUN. You can then use *PLD to load the compressed file. Also *PSV can be used to save a screen in compressed format. Examine the file 'GALLERY' to see this in action.

If you have drawn a picture which you would like others to see then send it in to us on disk. If we think it is good enough we will include it in our Gallery and give a reward. So if you want to be famous (and a little richer) then start painting. Details of how the picture was put together and full name and address plus telephone number would be appreciated. Ensure that your disk is clearly marked as well. Send your disks to our editorial offices (the address is at the front of the magazine).



STYLISH PRINT

AUTHOR

DOV ROSNER

FILES

'DEMO' .....BASIC
'T.STYLE' .....TEXT

STYLISH PRINT is a machine code routine which provides three new fonts on the BBC micro. The utility is ideal for creating professional front ends to games and applications. The routine works in all screen modes (except MODE 7) and is extremely easy to use. All fonts are available at the same time and therefore you can mix font styles and any colours (limited only by the screen mode you use) on the same line to achieve variegated results.

The routine uses only legal calls to the BBC operating system ROM and therefore can be used on second processors as well as co-processors and shadow RAM systems. It was tested successfully with BASIC 1, BASIC 2 and BASIC 4 and with a Solidisk FourMeg shadow board.

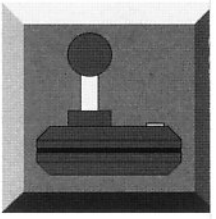
Three new fonts are provided italic, bold, and thin. These fonts are transformations of the standard BBC font hence no data tables are required, and as a result the size of the machine code program is less than 200 bytes. Therefore the routine does not suffer either from memory space taken by the character set, or from the restriction of using one font at a time.

The fonts are toggled through VDU codes. The codes are:

VDU 252: standard text
VDU 253: thin font
VDU 254: bold font
VDU 255: italic font

For the full article see disk B of *Fast Access*.

GAMEPLAN PART II



AUTHOR

PETER SCOTT

FILES

'GLOSS' ..... BASIC
'SCREEN' ..... DATA
'T.GAMEPLN' ..... TEXT

SURFACE GLOSS

Presenting a program can make the difference between it being taken by a software company and it being rejected. It has little effect on the game itself, but leaves a large impression on the people playing it.

Presentation involves many things. Redefining the character set is a good start. Most users see the standard set all the time and a new character set that fits in with your game creates a good impression. Try and make it easily readable, especially any status displays that may have to be read quickly.

I try and avoid using the VDU23 redefinition of characters, as this is slow, unwieldy and uses page &C on a standard BBC but not on a Master. A game like 'Omega Orb' or 'Ransack' uses the game sprite routine to print a smaller character set which is half the width (4 not 8 pixels) to give 40 columns in mode 5.

Some form of demo mode is nice, but this may be impractical and can take a lot of memory. A tune and high score table are usually sufficient enough. Small touches such as an abort key are important to players and take little memory or effort, and you should include them from the word go.

Pause/restart and sound on/off or volume controls are the two other options that should be compulsory on any game you write. Make all these keys respond quickly.

Make all lengthy begin/end graphics sequences 'abortable' : meaning they will end and the game will continue if you press a key.

Many people would mention skill levels and screen designers when it comes to finishing a game off : they are easy to include (you'll probably have a separate screen designer anyway used in the creation of the game).

User-selectable skill levels should be avoided, as they are simply a method of disguising poorly thought out difficulty levels. Your game should start easy (why should anybody spend ten quid to get stuck on the first screen of a game?) Make the difficulty level gradually rise, and make it as tough as reasonably possible to complete.

Some form of level jump feature is an excellent idea, but don't make it too easy to get to the last level or screen of your game, as this ruins the enjoyment of virtually all players. Passwords on completion of a level are a good idea.

Redefinition of keys is a good idea, especially in a game such as Superior's 'Barbarian', with two players playing at once.

Try and include a hidden cheat mode that is almost impossible to find by accident. This makes it easier for you to complete your game and also allows reviewers to get further into your game than they might otherwise get.

Jazzy displays and fancy screen clear routines all help in the presentation of your programs. This month's demo program includes many of the features you can include in your own programs. CHAIN "GLOSS" to see them or select the GLOSS icon from the menu.

All these features aid in presenting your game, but take up precious memory. If the choice is between an extra level or better presentation, it isn't too simple to decide which to do. If you have many levels in your game, put in the presentation. If not, don't. Remember that surface gloss adds to your basic game : it is no substitute for a decent game.

GOLDBANK

AUTHOR

RICHARD HOLMES

FILES

'GB-LOAD' .....BASIC
'GB-CODE' .....M/CODE
'GB-CHAR' .....DATA
'GB-GAME' .....BASIC
'T.GOLDBNK' .....TEXT

The object of this fast action game is to protect your bank from bandits and take as much gold from depositers as possible.

This simple and addictive game makes use of MODE 8. This is a low resolution but colourful screen mode which is produced using clever programming techniques. It gives all 16 colours on screen but takes only half the normal memory normally required for a full colour screen.

Instructions and game controls are all presented on the screen at the beginning of the game if you need them. So to start the game just select Gold Bank from the menu.

LET US KNOW !

We would like to know what you think of your new disk magazine, so let us know about it.

- Is there a program which you think is excellent.
- Do you think that there should be more of a particular type of program.
- Do you need more information.
- Are having compatibility problems with a program.
- Do you have something to say about Fast Access in general.
- Do you have any bug fixes or improvements to published programs.

Write To:
FAST ACCESS
6C Belgic Square, Padholme Road,
Peterborough.
PE1 5XF.

Free Disks !

All you have to do is persuade a friend or colleague to subscribe and we'll reward you with £15 worth of ***SpriteSytem and Biochip*** the super all-in-one sprite creation package. *Biochip* is a great game created using *SpriteSystem*. This two disk package is fully compatible with all the BBC range, including Model B/ B+ Master and Electron.

Just get your friend to pop his or her application in the post with a cheque for £42.00 payable to A.S.P. Access/Mastercard & Barclaycard VISA orders also taken, so include all the necessary details. Make sure he or she includes <u>your</u> name, address, subscriber number and whether 40 or 80 Track disks are required.

To ensure you get your **Free Gift!** <u>all</u> applications should be sent to the following address:

Fast Access Free Disk Offer!
C/O Mark Webb, ASP ArgusHouse,
Boundary Way, Hemel Hempstead,
Herts.HP2 7BH

JUMBLE

AUTHOR

D.J.HEALY

FILES

'JUMBLE' .....BASIC
'T.JUMBLE' .....TEXT

Jumble is a test of skill in sentence construction. A simple sentence is broken into individual words and then jumbled up. The words are displayed on the screen in a stack. To select a word you press the SPACE bar. This moves the white highlight onto the next word in the stack. When you want to place a word in the sentence you simply press <RETURN>. The object is to select and place the words in the stack in the correct order that they would appear in the sentence. When all the words have been placed the computer will tell you whether or not the sentence has been correctly put together. If it is correct then the next sentence is displayed. If it is wrong then the same sentence is shown again. The program will end when 26 different sentences have been sorted out.

YOUR
DO
A
MUM
YOU
LOT ?
HELP

DO YOU HELP YOUR MUM A LOT ?

The above is an example of one of the many mixed up sentences which have to be sorted. Below it is the finished sentence.

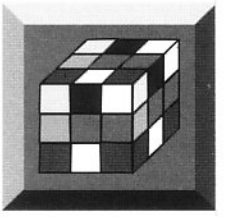
Fast Access Disk Forum

Welcome to the Fast Access Disk Forum. This is the place where you can share problems or solutions with other subscribers. Whenever there's space we'll carry mini reviews and news items You can also share your opinions on anything to do with Fast Access here. In short a forum where anything goes so long as it's relevant.To kick matters off I would like to welcome all our subscribers, both new and those from Disk User days. I must also apologise for the delay in sending out the first issue. Production problems were to blame. Hopefully by the time you read this we should be back on schedule, and you should also have received your Wordpower shareware disk.The first known problem will have been encountered by Model B/ B+ owners only when trying to run Ewen Setti's Blastout program. The channel or other error which halts the program can be overcome by first hitting <ESCAPE>, then typing RUN at the basic prompt. The problem was due to the sprite data being downloaded to an area of memory used by the DFS for storage of variables. We have now moved the sprite data to another location, and the problem should no longer occur. Don't try and modify the menu by copying over the version from this issue as it's a complicated business, and you could cause more problems than you solve.

Wordpower

We have used this shareware wordprocessor as an extra incentive for people to take out a Fast Access subscription. So now many of you will have a copy. But is it sufficiently better than your present word processor to replace it completely. Broadly the answer to that question is yes, but with a few caveats. The program is faster than View in 80 column mode although not as fast as Wordwise in 40 columns. Model B owners will need the ROM upgrade, otherwise they should stick with Wordwise. The control key sequences are generally easy to understand, although the terminology used is a bit strange. I would have liked an undo command and some onscreen menus. On a Master 128 the program really sizzles, and using ADFS you can overcome the limitation caused by rather too many files on the original system disk.

KNIGHT'S MOVE



AUTHOR

A.EVERITT

FILES

'KN-LOAD' .....BASIC
'KNIGHT1' .....BASIC
'KNIGHT2' .....BASIC
'T.KNIGHT' .....TEXT

THE most interesting piece in the game of chess is the knight. The character is based upon a cavalry officer riding his horse. Unlike all the other characters in the game, the knight does not move in a straight line; instead he moves in an 'L' shape, one square in one direction, horizontally or vertically and then two in the other. Thus if he is in the centre of the standard eight-by-eight chess board,he has eight possible moves.

I recently came across a puzzle in which you had to see how many moves a knight piece could make, the only condition being that he could never land on the same square twice. With this in mind, I wrote two programs. The first one allows you to solve this problem, starting from any point on the board. The second program will work out the answer for you.

THE PUZZLE

To move the knight press a numeric key between one and eight. This will move you in the direction shown on the compass. If you make a mistake or can not go, then you can take back your last move by pressing 'T'. If you want to start again from a different position, press 'R' and answer 'Y' to the message that appears at the bottom of the screen. Note that the current position is shown in yellow as opposed to red.

THE SOLVER

Pressing a key will manually prompt the computer to move. When it cannot move any further it will stop and will tell you to press a key. The beep is to inform you that the best

route so far has been found.

The computer will try to find the best route in a similar way to you, but will rely totally on logic. For each move, it will first try direction one. If this is not possible, direction two and so on. Now if you press a key, the computer will try another route. It does this by taking back moves until it finds an alternative move . Once it has found a new move,it will continue to move as before. This way, if the computer is left for a long enough time, it will check every possible route.

Function key 0 sets slow mode.

Function key 1 sets fast mode.

Function key 2 sets ultra-fast mode.

Function key 3 sets non-stop mode. Unlike all the other modes, this mode does not stop until told to do so (by pressing any key). This is the fastest mode as the only delay is caused by updating the table.

When in either the ultra-fast or the non-stop modes, the grid is blank. Thus when you go from the slow or fast mode to one of these modes, the grid is blanked. When you return to the slow or fast mode, the current route is redrawn.

Function key 4 graphically shows the best route taken.

Function key 6 will save the best route and the current route so that a session can be split without having to start again.

Function key 7 will print the moves involved in the best route on a printer.

Function key 9 allows you to restart with new start coordinates.

You will probably have noticed that I omitted function key 5 from the list. If you want to load a file, you should press function key 5 when initially asked for the X-coordinate. If you are unsure of the filename, enter a question-mark instead; this will call up the current directory.

Although the program would take a long time to check every single route, it can usually beat the best human effort in a few dozen attempts.

Happy solving...

SORT TUTORIAL PART II

AUTHOR

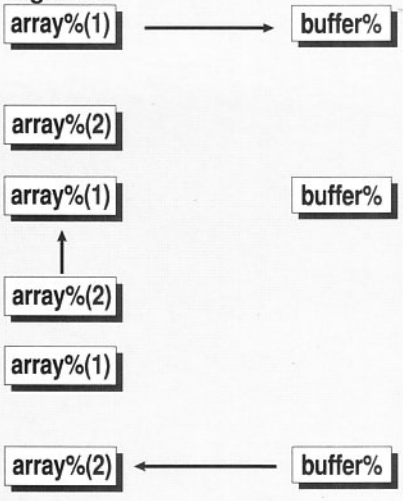
MOSTYN HELLARD

FILES

'TEACH2' .....BASIC
'T.SORT2' .....TEXT

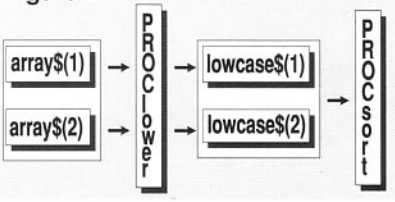
Here is the second and final part of the sort tutorial. This issue's article refers to three figures These can be found on the following page. The figures on this page are a graphical illustration of last issue's sort technique. Please refer to the relevant text for an explanation.

Figure 1



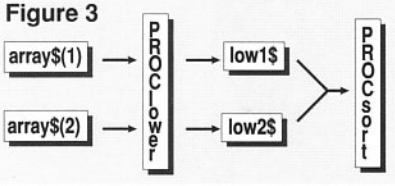
First, The contents of 'array%(1)' are placed in the temporary storage variable 'buffer%' Second, the contents of 'array%(2)' are copied into 'array%(1)' so that both elements contain the same value. Finally, the contents of 'buffer%' are copied into 'array%(2)' reversing the original values

Figure 2



Two separate arrays, the data is initially entered into 'array$'and is processed by the procedure 'PROClower'. The array 'lowcase$' is parallel to 'array$' and receives the lower case data. The sort routine then carrys out it's comparisons on 'lowcase$' only but performs it's swaps on both arrays.
NOTE: The array is processed by 'PROClower' once only.

Figure 3



This time only the one array is used, the data is again input into the array; 'ar-

ray$', each time a comparison is to be made the routine takes elements it requires from the array, processes them with 'PROClower', inserting the results into the temporary storage variables 'low1$'. Having done this the routine compares the new variables and if a swap is required it carries out the normal procedure on the array; 'array$' involving the variable 'buffer$' (see figure one).

Figure One: An example of a shell sort routine that sorts the array; 'array%', that is 'length%' elements in length.

```
1000  DEF  PROCshell
1010     interval% = length%
1020     REPEAT
1030        interval% = interval%
DIV 3
1040        IF interval% = 0 THEN
interval% = 1
1050        REPEAT
1060          flag% = 0
1070          FOR element% = 1 TO
(length% - interval%)
1080             IF array%(element%)
>  array%(element%+interval%
                 THEN PROCexchange
(element%,interval%):flag% = 1
1090          NEXT element%
1100        UNTIL flag% = 0
1110     UNTIL interval% = 1
1120  ENDPROC
2000  DEF  PROCexchange
(element%,interval%)
2010     buffer% =
array%(element%)
2020     array%(element%) =
array%(element% + interval%)
2030     array%(element% + inter-
val%) = buffer%
2040  ENDPROC
```

Figure Two: Data table for shell sort listing.

interval% **Integer**

Contains the difference (in elements) between the two items to be tested.

length% **Integer**

Total number of elements in the array; 'array%'.

flag% **Integer**

Flag variable that indicates whether a variable exchange has taken place in any given pass (1 = yes : 0 = no).

element% **Integer**

Number of the highest element to be tested or exchanged.

array% **Integer Array**

Array of length; 'length%' elements to be sorted.

buffer% **Integer**

Used as temporary storage during the variable exchange process.

Figure 3 The effect of different sort routines on time taken to sort an array