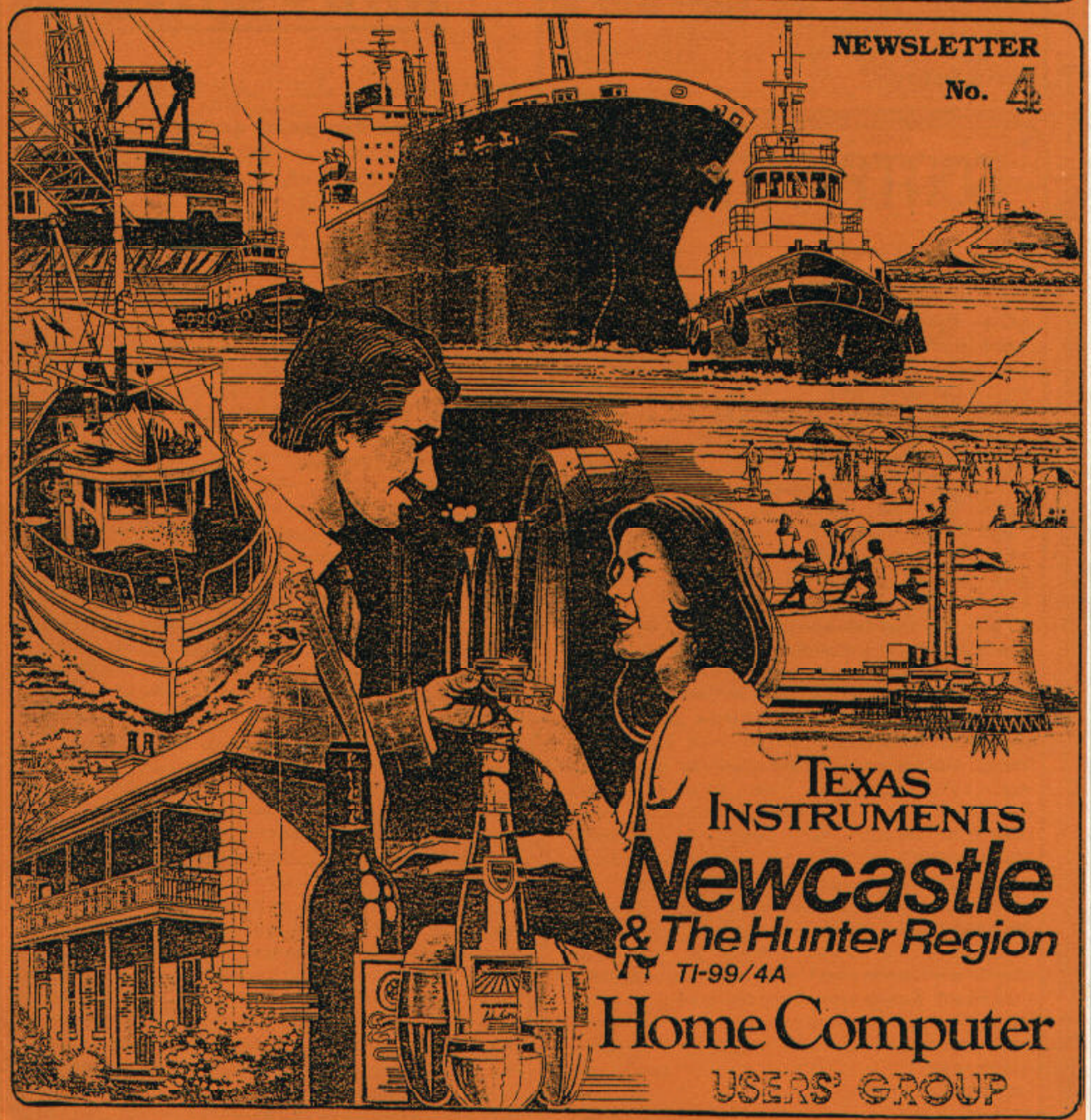
Sept 85

HUNTER VALLEY 99'ERS NEWS

TI 99/4A

HOME COMPUTER NEWSLETTER



NEWSLETTER No. 4

TEXAS INSTRUMENTS

Newcastle & The Hunter Region

TI-99/4A

Home Computer

USERS' GROUP

DISCLAIMER

THE HV99'ER NEWS IS THE OFFICIAL NEWSLETTER OF THE HUNTER VALLEY NINETY NINERS USER GROUP. WHILST EVERY EFFORT IS MADE TO ENSURE THE CORRECTNESS AND ACCURACY OF THE INFORMATION CONTAINED THERIN, BE IT OF GENERAL, TECHNICAL, OR PROGRAMMING NATURE, NO RESPONSIBILITY CAN BE ACCEPTED BY HV99'ERS NEWS AS A RESULT OF APPLYING SUCH INFORMATION.
TEXAS INSTRUMENTS TRADEMARKS, NAMES AND LOGOS ARE COPYRIGHT OF TEXAS INSTRUMENTS.
HV99'ERS IS A NON PROFIT GROUP OF TI-99/4A COMPUTER USERS, NOT AFFILIATED IN ANY WAY WITH TEXAS INSTRUMENTS.

CONTRIBUTIONS

You are invited to contribute copy for publication in HV99 NEWS.
Copy for publication may be typed, hand written, or submitted on tape/disc media as files suitable for use with TI-WRITER (ie. DIS/FIX 80 or DIS/VAR 80). A suitable PUBLIC DOMAIN word processor can be supplied if required from the club librarian.
Please include in your copy sufficient information to enable the file to be read by the EDITOR eg. FILE NAME etc.
The preferred format is 36 columns and page length 66 lines, filled and right justified.
All copy printed in HV99 NEWS is considered to be PUBLIC DOMAIN.
Other user groups wishing to reproduce material in HV99 NEWS may do so as long as source and author are recognised.
Copy for publication can be submitted to.

THE EDITOR
HV99 NEWS
15 GAYTON CL.
WARNERS BAY 2282
NEWCASTLE

General address for all other club related correspondence.

THE SECRETARY
HV99'ERS
25 RESERVE RD.
WANGI 2267
NEWCASTLE

YOUR COMMITTEE FOR 1985

A. WRIGHT	PRES.	PH. 468120
P. COXON	SECT.	PH. 751930
B. RUTHERFORD	TRES.	PH. 496184
A. LAWRENCE	LIBR.	PH. 486508
S. TAYLOR	EDIT.	PH. 487078
B. WOODS	EDAS.	PH. 662307
T. MCGOVERN	TECH.	PH. 583182
B. MAC. CLURE		PH. 437431
D. WINTON		PH. 591882
G. JONES		PH. 573744

NOTICE BOARD

Richard Terry's Forth Interest Group has now reformed and will be meeting regularly on Thursday nights. If you require further information you can contact Joe El Presidente Wright most evenings on 468120.

At the last committee Brian Woods was nominated as assistant editor/sub librarian. What this in effect means that in future all newsletters etc received by the club will go directly to Brian who will then sift the wheat from the chaff, catagorise, homogenize and then present them for general distribution. Brian will also be taking charge of all the clubs reference books and manuals. The same borrowing system will still remain in force ie. just sign the book.

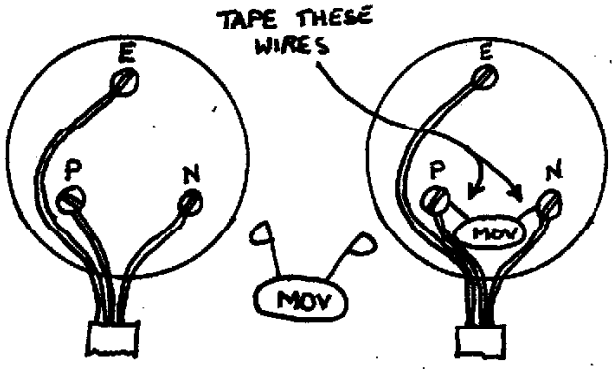
Alan Byrne has resigned from his position on the committee due to his extremely heavy work load at the present time. At the last committee meeting Garry Jones was elected to take Alans place. Welcome aboard Garry.

An invitation has been sent to the Newcastle Microbee User Group inviting their members to attend one of our general meetings in the new year. Hopefully this will result in a reciprocal invitation so that we can see what those guy's can get up to on their Aussie built machines.

The 32k. expansion project should be off the ground shortly with most of the hardware already purchased. If you have your name down and havn't paid yet you had better contact Brian ASAP or you will miss out.

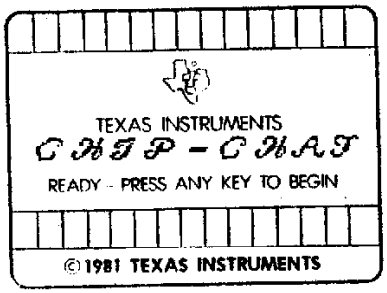
SECRETARY"S NOTES BY PETER COXON

Well another month has gone by and I'm hacking the keys once again. Things are really starting to hot up down at H.Q. The group is going from strength to strength and building up a great reputation for the user's of a defunked machine. Just like to say he.lo to all our new members. Now down to business. The chips for the 32K matchbox expansion have arrived from the West, thanks to Bernie Elsner. We have ordered some more for the late comers, together with a few spare, so if you have decided to boost your machine up and have'nt ordered the kit yet, contact me A.S.A.P on 751930, as first come, first served. The boards have also arrived, so an evening with a soldering iron is being organised, and those who have ordered will be contacted when a date is fixed. The club's RS232 will be available (fingers crossed) by the end of the month, and available for loan as soon as a loan system is worked out by the Committee. Also to go with the above mentioned, we have a small Committee looking into costs etc of printers for the group. Their recommendations will be put to members for approval, so hopefully by the end of October we will have a printer to loan after the Editor has finished putting the Magazine together. Bob MacClure has donated some of his generous time and agreed to become our module Librarian, we only have 3 at present, but as more become available we will endeavor to purchase them. Donations will be gratefully acknowledged! For more info you can ring Bob on 437431. Another member who has joined the ranks is Brian Woods, he is now our Sub-Editor and magazine dissector. The reason for this is to take some of the weight off our Librarians shoulders, he will try to get all the best bits of info out for our magazine, then hand them over to Al for loan. Another purchase we have arranged this month is a spare 99/4A computer. This of course is for use at the meetings, and available for loan if yours breaks down! The *LOGO* competition is now closed and there are three finalists, they are Scott Johnson, Audrey Coxon and Albert Anderson. A big thank you to all thosewho entered, a winner will be announced shortly. I must remind you to check out Al's Library news for some exciting news. Also a get well message to Bob MacClure's son Darren, who is laid up at present. Our best wishes for a speedy recovery go to him, from all HV99'ers everywhere. A little tip for all cassette users who have trouble with their power supplies, i.e., surges/spikes, to buy a protection for these troubles can cost anything from $20 to $100's, but if you buy a metal oxide varistor 220 to 240V Siemens model SI0V-520K230 or equivalent from your local Electronic's store, all you need to do is open the back of the mains power



plug, connect pos to pos+, neg to neg-, tape the bare wires on the M.O.V and re-assemble plug. This should remedy most of your problems, ifnot you might be looking at a more advanced system, a disc system has a similar device already built in. I cannot take any responsibility for any modifications that could cause damage to you, or your machine. Another item just to end off with, if you want to connect a printer direct to your machine, but do not want to up-grade with an expansion Box or a Smart Cable at a cost of $160, hold out for another month longer as we might just have the answer you are looking for! Out of memory - See you all at our next meeting.

Peter C.



****
* PROGRAMMING TIPS *
****
These two programming gems come to us from the "NORTHWEST OHIO 99ER NEWS", many thanks to Ken Sheets of that group.

1.With this routine all you do is push the Fire Button on either joystick, and the computer will remember which joystick you are using.

```
90 PRINT "PRESS THE FIRE BUTTON ON
THE JOYSTICK YOU ARE USING"
100 CALL KEY (1,J1,STATUS)
110 CALL KEY (2,J2,STATUS)
120 IF J1+J2<>17 THEN 100
130 JS=INT(J1/18+J2/9+1)
140 CALL CLEAR
```

2.If you are using a black and white television, adding the following statement at the beginning of your programme will disable the colour generating circuit in the TI99/4A. This removes the pattern of vertical lines often seen on black and white TV's, and also increases the sharpnes of the characters on the screen.

```
100 CALL SCREEN(15)
```

VIA. TIUP

****
* SAVING TIME IN FORTH *
****
When loading routines from the system disk, I have found that I need to remind myself that it is possible to chain a series of commands together, ie;

-EDITOR -PRINT -FLOAT <ENTER>

This reduces the time spent waiting for the cursor to reappear. Likewise you can reduce the number of disk accesses by chaining your commands when going from EDIT to LOAD, ie; on leaving screen #2.

FLUSH 2 load <enter>

Steve Wilkinson. TIUP.

****
* SPEAK OR NO SPEAK *
****
Here is a handy hint from the "St. Louis 99'ers" for those of you with speech synthesizers. It is possible for a programme to detect whether or not a speech synthesizer is connected to the console, and thereby design your programme to use speech or otherwise as is appropriate. Try using the following format in your programmes.

```
100 CALL PEEK (-20672,SP)
110 PRINT "HELLO HOW ARE YOU"
120 IF SP=0 THEN 130 :: CALL SAY
("HELLO HOW ARE YOU")
```

Line 100 returns 96 if the speech synthesizer is attached, and 0 if it isn't.

VIA. TIUP.

****
* EXTRA LONG *
****
The 9900 Users Group Inc of Moorstown, New Jersey, offers a suggestion that may be of benifit to those who write lengthy pieces of text in PRINT statements in Extended basic. What it allows you to do is to eliminate the discontinuous appearance of the text when the programme is run. It may also have applications in programme lines.

First, begin by typing in the PRINT line. As you reach the end of the line length limit, enter the quote, type in a double colon statement separator (::), press ENTER and start another PRINT line on the next programme line. When you run the programme you will notice that there is no break between the first PRINT entry and the second. You will need to experiment a little to determine just where the statement separator should be placed. When it is done properly, the end of the first line is followed directly by the beginning of the second.

****
* HELP WANTED *
****
We have a number of projects and future magazine articles in the melting pot at the moment, but unfortunately have had to turn the burner down to low due to the lack of time available. If you would like to help contact someone on the committee, we will be more than grateful for the offer.

TI FORTH INT. INFORMATION CENTRE

The address given for the FORTH INTERNATIONAL INFORMATION CENTRE has now been changed. The new address is listed below.

TI FORTH INTERNATIONAL INF. CENTRE
4122 NORTH GLENWAY,
WAUWATUSA,
WI. 53222
U.S.A.

NEW SCOTT ADAMS ADVENTURES FOR TI

Tex-Comp the large American mail order software company has contracted Scott Adams the author of the original TI Adventure module series to convert his four latest adventures to run on the TI.

The titles include the Adventures of Buckaroo Banzai, the Incredible Hulk, Spiderman and the Sorcerer of Claymorgue Castle.

TI99/4A OR TI99/4QI ?

Which do you own? Let me explain. In the final months of production of the TI99/4A, TI found itself in a toe to toe price war with several other computer manufacturers, the only way to lower the production costs of the machine was to lower the number of parts that went into it. TI went about this by integrating a number of chips into a few chips that would do the same job. The circuit board layout was also modified to assist in quicker assembly. TI designated the updated machine as Quality Improved (QI).

The easy way to tell if you own a TI9/4A or a TI99/4QI is to look at the I/O port (where the speech synthesizer plugs in) if the metal RF wiper is a copper colour you own a TI99/4A, if it is a silver colour your's is a TI99/4QI. In either case there is no cause for alarm as both mchines are electrically identical.

CLUB LOGO COMPETITION

The club Logo competition is now closed with the entries being viewed at the last committee meeting. The winning entry will be announced soon. The committee wishes to thank all those who contributed entries to the competition.

EXTENDED BASIC 1V.

The MYARC company of America has just announced the release of "EXTENDED BASIC LEVEL 1V."

The new command module based language is reportedly 100% compatible with TI Extended Basic and features windowing, 40 character text display and is claimed to be up to three times as fast as Extended basic. Other features include graphic commands such as draw, circle, fill and rectangle as well as other niceties such as improved error handling routines and integer variables.

The new version of Extended Basic will comprise a module and disk. Unfortunately it requires the use of MYARC'S 128K card to operate.

NEW PEB CARD FROM Cor Comp

The new card comprises three utilities. A clock/calendar, 64K printer buffer and an inbuilt edge connecter for fitting the circuit board out of your speech synthesizer. Cor Comp call their new card the Triple Tech.

NEW TI99/4A BASED MAGAZINE

The new exclusively TI99/4A based magazine called MINI-MAG 99 is produced in America by S.O.S. Publishers of California. So far I have only had the opportunity to read Volume 1, No. 3 which in my opinion was not quite up to the standard of MICROPENDIUM, this is not to say that that given time it may not develop into quite a worthwhile alternative. Certainly there is a market for a new magazine amongst the information starved TI99/4A diehards.

Foreign Subscription rates are $US.35.00 (seamail) and are obtainable from.

S.O.S. PIUBLISHERS
21777 Ventura Blvd.
Suite 203.
Woodland Hills,
California 91364.

The issue that I read contained a particularly informative article on TI-WRITER file management, as well as articles on FORTH, and Plotting on the TI99/4A using the RADIO SHACK (TANDY) CGP-115 PLOTTER.

TI-WINDOWS

Several new Application Software packages on the market for the TI99/4A are claiming the use of windows. The most interesting of these new programmes is one which allows you to write windowing into your own programmes, it is produced by the CSI DESIGN GROUP from the U.S.A.

Windows allow the user to change modes of operation without really thinking about what we are doing. A mode is a part of an application that the user has to formally enter and leave, and that restricts the operations that can be performed while that particular mode is in effect. Modes are a part of most applications. For example when running TI's Editor Assembler there are 3 specific modes: Edit, Assemble and Execute a program (either through Load and Run, Run or Run Program File). Normally when you are in one mode there is nothing you can do to invoke a part of a different mode.

This is not all bad but modes can be confusing, especially when you are in the wrong one. Windows and pull down menus (which are actually windows in their own right) simplify this. If you have disk operations in one window and an editor in a second, you switch modes simply by selecting the window that has the mode you wish to be in.

So what does this mean to me? Well no one can be forced to write programs containing windows. But it is a reality in the computer world that application that are not friendly and easy to use(read modal) do not sell as well or become as popular as those that are friendly. Intelligent users are more impressed by non-modal applications and like being given choices. Another fact is that a friendly application that you have written is much more satisfying (not to mention ego building) than a modal application.

The CSI WINDOWS programme takes the burden out of writing TI applications that use windows, by developing a group of subprograms that are clear and usable as well as memory efficient and fast.

You do not need to know assembly language to use CSI WINDOWS! The subprograms are accessible from three languages: TI Basic (requires Editor Assembler or Mini-Memory Cartridge), TI Extended Basic, and TMS9900 Assembler. All languages require a minimum of 48K of Ram. All the subprograms are available from every language and all the subprograms called from the both parameters use regular TI parameter passing.

CSI WINDOWS is written in super-tight assembler and uses only 8K. In this 8K are an assortment of primatives that allow you to perform many operations on and in windows.
The programme has support for up to 5 windows, each having dimensions of up to 256x192 pixels.
Anyone wishing to purchase a copy of the programme should send $US 24.95 plus $US 5.00 for post and packaging.

CSI DESIGN GROUP
BOX 50150
St. LOUIS,
Mo.63105

AXIOM INTERFACE

I noticed in the last club newsletter that Peter coxon was asking for information on the AXIOM PARALLAX PRINTER INTERFACE.
On my particular Axiom I have fitted a sub miniature SPST switch on the top panel, right hand end, dead centre and wired it up to the jumper option (J1) and suitably marked <PIO-RS232< so that either device name can be selected with the flick of a switch; also one other peculiarity has cropped up, sometimes and that is if for any reason the printer is operating and is interrupted either by a programme fault or the operator halts the printing run with clear (FCTN 4) then it is very advisable to type >close #< (whatever file No. was opened). Before the printer is switched off because sometimes the interface will generate a signal that causes the computer to quit and return to the title screen. I have yet to investigate this oddity.

Regards.
Ron Kleinschafer
Grawin
Via Walgett. 2832

THE CORCOMP 9900 MICRO-EXPANSION SYSTEM

SYSTEM REVIEW

BY

BRIAN WOODS HV88.

A few months ago I let my head (and my purse-strings) go and purchased the then just released Corcomp Expansion System. After managing to learn to drive it I am now in a position to write about what it can do.

The system is a compact "box" about twice the width of the TI Speech Synthesizer and as high as the console and plugs directly into the side of the computer where the PE Box interface goes. The system comes with an RS232 card with serial and parallel ports and 32K memory which contains a double sided, double density disk contoller and allows up to 4 DSDD drives to be used. Because of the size of the system it makes a more "transportable" system than TI's PE Box. One disadvantage with this system is that the "expansion box" and all drives must be externally powered so now I am the proud(?) owner of 3 power transformers (1 each for the computer, expansion system and the disk drive).

Once the system is attached and powered up, the first difference is the main screen - no longer the now familiar TI screen but a less fancy controller screen.

```
        9900
   DISK CONTROLLER

  PRESS        FOR
  1. DISK MANAGER
  2. TI BASIC
  3. EXTENDED BASIC
  4.

SPACE BAR FOR COLOR BAR
```

With the CC System, the disk manager is supplied on disk and is considerably faster than TI's Disk Manager module. With the Manager disk in drive 1 and selecting option 1 the Manager auto-runs. The first screen to greet you is the main menu:

```
   SELECT OPTION: 1

1. FILE UTILITIES
2. DISK UTILITIES
3. DISK TESTS
4. CONFIGURATION MANAGER
```

When using the CC system for the first time it is probably best to go straight to Option 4, the Configuration Manager. This allows you to configure the Manager to your disk system. You select the number of tracks per side, number of sides and density for each of up to 4 drives (if you should be so lucky!). You also select the printer type, screen color and text color for the display. The second "page" of this option allows options for Interlace (position of sector within track), Turbo Option (turns off Verify after Write command) and then asks if you wish to save this configuration to the master disk. If you press "Y" then it is saved and becomes the default option whenever you run the manager.

Option 1 - File Utilities

On selecting this option, then option 1 from the Utilities screen, a catalogue of the disk in the drive is displayed. If a full DSDD disk is used it would be impossible to show all the files on the screen at once because a full DSDD disk can hold 360K or 127 Basic programs of 200 lines each. The CC displays the catalogue by the screenfull and screens can be "turned" using CTRL X and CTRL E. This utility allows you to copy files from one disk to another, delete a file from the disk and change the protection all in one operation. To the left of the filename is the default letter "N". By changing it to "C" it copies the file to another disk, "M" copies it to another disk and deletes the file from the original disk and "D" deletes the file. The filename can be altered by typing over the old filename and the protection can be altered by typing "P" to protect and "U" to unprotect. As each command is being carried out a message on the screen tells what is happening.

The second option under File Utilities is "Load and Run" and lets

you load Assembly Language DIS-FIX files without using the Editor/Assembler module.

Option 2 - Disk Utilities
When this option is selected there are 4 functions that may be performed.
The first is a catalogue of the disk which shows filename, size (no. of sectors) and protected/unprotected status. If necessary screens may be "turned" as above and a printout of the screen can be obtained by pressing FCTN Ø.

The second option is to copy the entire disk. The major problem with this function is that it is virtually impossible to copy single density to double and vice versa because the drives normally used on a home computer cannot change densities quickly enough so an "error" is detected and the copying aborts. When copying SD to SD or DD to DD there are no problems and the copying proceeds . Imagic Sydney say they have had quite a few phone calls and warranty claims because of this "fault" until told by a technician that the problem is with the quality of the drives and not a system fault, so it's something we have to learn to live with.

The third option is to rename the disk.

The final option on the Disk Utilities screen is to initialize a disk. The disk can be initialized in any format you wish :- SSSD,SSDD,DSSD or DSDD. You are also given the option of having the Manager installed on the new disk if you wish, which can be useful on utility type program disks, but uses too much space on a SSSD disk - the manager uses 96 sectors itself!

Option 3 - Disk Tests
This feature allows you to test disks for errors and has a read only or destructive test options

These, basically, are the functions of the Disk Manager of the Corcomp system and after a bit of practice and experimenting the commands are quite easy to master. Apart from limitations with the copy disk facility everything operates smoothly and is quite "user friendly".

As well as the Manager on the disk there are new Assembly Language commands and program statements available on the disk but as yet I have not made much of an attempt to use them. They include CALL POKE, CALL PEEK, CALL POKEV (to read and write to VDP RAM), CALL MOVEM (for moving blocks of memory from one location to another - it can move about 30-40 screens full of data per second!), CALL EXEC (for executing ROM or expansion memory routines) and CALL WRTRG (loads the VDP Write Only registers). All of these may be used in either BASIC or EXTENDED BASIC and reside in the 9900 Disk Controller Interface. By using the command CALL MGR in XB the Manager can be loaded directly from the disk rather than "QUIT"ting and going through the master screen again. All of these new commands/statements are fully explained in the excellent manuals that come with the system.

The first manual covers the use and operation of the 32K Memory Expansion and the Disk Controller. It details the setting up of the system, detailed information on the nature, use and care of diskettes, a detailed explanation of files, how to use the Disk Manager and how to use the new commands/statements that reside in the Disk Controller Interface.

The second manual explains how to use the RS232 card. It shows how to set the system up for your printer and modem. Both manuals go into fairly explicit instructions on all facets of the system so that by working your way through the manuals the operation of the system becomes quite straight-forward.

The Corcomp Expansion System is very easy to master, quite small compared to TI's PE Box but able to do all it could do and being capable of using double sided double density disks bring about noticeable savings in the number of disks (and money!) required compared to SSSD systems.

With the recent devaluation of the $A it has unfortunately pushed the price of the Corcomp system to $895 plus power transformer (about $50) then by the time you add even 1 DSDD disk drive it becomes quite an expensive proposition but, all in all, it turns a great computer into a compact, remarkable system that I would recommend to anyone interested in expanding their basic system.

LEARNING FORTH WITH RICHARD TERRY HV99.

EDITORS NOTE: THIS ARTICLE ORIGINALLY APPEARED IN ISSUE 3. UNFORTUNATELY DURING FORMATTING SOME OF THE TEXT WAS MISPLACED. IN FAIRNESS TO RICHARD TERRY AND HIS FELLOW FORTH FIENDS THE ENTIRE PRTICLE IS REPRINTED THIS ISSUE

PREAMBLE

It was with some disbelief I read of my so called offer in last months magazine, to write a regular FORTH column. Once I got over my surprise I realised that it was just Jo's way of saying DO IT. I must admit I did tell Jo I was working on a FORTH program and that perhaps it could form the basis of an article but a firm promise -NO.

So, I don't intend to be THE regular contributor. I'm sure there are many more out there in Forthland with a greater lengh of experience than my 6 weeks in the field, so if the inclination stikes you post the articles in.

Also my approach to FORTH programming will be personal- ie gleaned from my trial and much error experience. It will not necessarily be good FORTH. It will probably be excessively wordy, illogical and at times appear not to work. Please don't hesitate to send in better more concise versions as the pearls of wisdom flow from your grey matter through the keyboard to your FORTH screen, or ring me in reasonable hours on 2245Ø if somthing does not run.

HINTS.

First a few hints to speed you on your way:

1.MAKE MULTIPLE BACKUPS of your Master disk just in case.

(see Buffers below)

2.BUFFERS

-Understand the concept of how they work. There are 5 1K block buffers. You should always start working by using EMPTY-BUFFERS. The reason is as follows. Suppose you are EDITING multiple screens. Each new one you EDIT is loaded from disk into the next available 1K buffer. The first five are fine but if you edit a sixth the system will automatically flush an updated buffer back to disk to make room for this new arrival. Fine is you happen to have the right disk in the drive, but if your pretty thick and forgetful like me your just as likely to have changed disks somewhere along th line and have the wrong disk in the drive. Typing FLUSH or if the system automatically updates; may overwrite that precious information it took you two days to figure out, so be careful!

3.EXPERIMENT.

Not coming from a mathematical or computing background, and being an entirely self taught basic programmer, I found reading the Reference Manual like trying to translate a foreign language from first principle. The information you really want to know is conveniently not there - like the subtext in a play or novel, in this case they plainly considered it too evident to write it in.

Try the words to see what they do, even if you do not understand the manual. It really is fascinating to see what is happening in the "guts" of the machine. If you are anything like I was to start with you will be typing along fine, then all of a sudden some seemingly simple instruction will go wrong, poking a number into a never to be discovered spot in the bowels of the machine, resulting in either total system lockup, or treating you to either a Kaleidescopic technicolor display or a seemingly endless spewing forth on the terminal of the contents of countless adresses, a scenario which ends either spontanously as if nothing had happened, or with the user turning off the machine and re-booting in frustration. While this happened to me with decreasing frequency as time passed (I became quite sneaky about what I did and often pressed the keys with some trepidation) I jumped for joy when I mastered BSAVE AND BLOAD which made

rebooting quick and painless.

BOOTING IN BINARY

Dont ask me what Binary images are, (ask Tony MgGovern!) because I dont know, all I know is using them is fast. Insert the Editor/Assembler module and choose Option 3 and type DSK1.FORTH to boot the master disk.

1.First modify SCR# 72 if you use a parallel printer by putting the MASTER DISK COPY in Drive 1 and after loading your -EDITOR and -COPY from the master menu typing 72 EDIT. Change the RS232 etc in line 4 to ." PIO" and correct the typing error in line 5 to read PAB-ADDR exit the screen using F'n 9 and type FLUSH to place your changes to disk.

2.Initialise a blank disk by placing it in Drive 1 and typing:
Ø FORMAT-DISK
Type EMPTY-BUFFERS

3.IF YOU HAVE TWO DRIVES place your initialised disk in drive 2 and type 180 DISK_HI ! which tells the system you are using 2 drives. Place your MASTER COPY DISK in Drive 1 then type Ø 9Ø 2Ø SMOVE which will copy the first 2Ø screens (Ø-19) containing the error messages and the boot screens and the binary core of forth onto screens Ø-19 of your new disk.

4.IF YOU HAVE ONE DRIVE:
Place the MASTER DISK COPY in Drive 1 and type:
Ø DISK_LO ! then press <ENTER>
: GETBLOCK DO I BLOCK UPDATE LOOP ; then press <ENTER>
5 Ø GETBLOCK then <ENTER>
wait till loaded then insert copy disk and type FLUSH
Reinsert Master disk after each FLUSH.
1Ø 5 GETBLOCK then <ENTER>
Reinsert Copy disk and FLUSH.
15 1Ø GETBLOCK then <ENTER>
Reinsert copy disk and FLUSH
2Ø 15 GETBLOCK then <ENTER>
Reinsert copy disk and FLUSH
This places the Screens Ø-19 onto your new disk.

5.Place MASTER DISK COPY IN DRV 1
And type COLD then <ENTER>.
Choose the options you wish to use from the master list including one of the editors and load them:
Eg:-PRINT -GRAPH -VDPMODES -COPY
Next type in -BSAVE -EDITOR

6.Place your new disk with the saved screens in Drive 1 then Type ' TASK 2Ø BSAVE then press <ENTER>.
This saves all your above options on disk from SCR#2Ø onwards.

7.Next we must modify the BOOT SCREEN -SCR# 3. Type EMPTY-BUFFERS then 3 EDIT and make the changes shown below on the listing of SCR#3.You can type in whatever heading you like.
Press function 9 then type FLUSH to flush your changes to DISK

8.If you wish to change you disk name for identification with the manager just alter the first 1Ø characters on SCR# Ø Eg to BIN-FORTH and FLUSH.

9 Type cold to re-boot and see the difference in speed

1Ø.Finally make a back-up copy, tape over the write notch and store in a safe place!

Since this takes up so little space you can easily make a copy of this to use when developing programs on the same disk, so that when you make mistakes like me and have to re-boot, you won't blow your transistors waiting interminably every time you have to reboot and recompile your options.

```
SCR # 3
 0 ( WELCOME SCREEN )
 1 BASE->R HEX 10 SYSTEM  ( Clears screen)
 2 0 0 GOTOXY ." Booting... BINARY FORTH" CR 10 0202 C!
 3 ( QUIT off)  R 3 GOTOXY ." -EDITOR" R 4 GOTOXY ." -PRINT"
 4  R 5 GOTOXY ." -GRAPH" R 6 GOTOXY ." -DUMP" R 7 GOTOXY ." -VDP
 5 DEP" R 8 GOTOXY ." -COPY"
 6
 7 DECIMAL 20 FLOAD
 8 1 VDPMODE !              ( Into text mode)
 9 0 DISK_LO !              ( Allows EDIT/COPY on all SCREENS)
10 190 DISK_HI !            ( Set up for 2 single sided drives)
11
12
13
```

Now our exercise for the month. From the outset I would like to stress this is only one solution, not necessarily the shortest (in fact I Know its not) and not necessarily the best. Other versions using 2 other different techniques are available by phoning either Jo Wright or Keith Bruce. If when perusing this example you come up with any brainwaves, don't stand on ceremony, let us in on the secret.

FORTH INTEREST GROUP: PROJECT INTEGER ACCEPT UP TO 5 DIGITS Ø2/Ø7/85

```
FORTH SOURCE CODE:                          STACK EFFECTS     ACTION OF WORD
------------------                          -------------     --------------

Ø VARIABLE TEMP 5 ALLOT                     (     --      )   Allot 7 byte buffer
: 3DUP      OVER OVER >R >R >R >R           (n1n2n3--         Copy top 3 Stack items
            DUP R> R> ROT R> R> ;            n1n2n3nn1n2n3)   n1,n2,n3
: PROMPT    Ø Ø  GOTOXY ." Number:" ;(     --      )         Put up prompt
: REPRINT   Ø 1Ø GOTOXY ." Voila!:" . ; ( --      )         Reprint number left on
                                                            Stack by CONVERT
: NOT       3DUP ROT 32 HCHAR ;             (ND,col,row--     Erase incorrect input
                                             ND,col,row  )  (see comment bleow)
: ENTER     DUP TEMP SWAP EXPECT ;          (ND  --  ND  )    Accept ND Characters
: ?DIGIT    DUP >R TEMP SWAP Ø DO           (ND  -- flag )    Check all,leave a 1 flag
            DUP >R C@ DUP 48 < SWAP                           if non-digit or a Ø flag
            57 > + R> 1+ LOOP DROP                            if digit and add flags
            R> 1 - Ø DO + LOOP ;
: CONVERT   Ø Ø ROT 1 - (NUMBER)            (Adr -- DN    )   Convert contents of Temp
            DROP DROP ;                                       to number leave on stack
: ACCEPT    3DUP GOTOXY ENTER ?DIGIT (ND Col Row--            Put up prompt accept
              IF NOT MYSELF ELSE DROP                  DN)    entry if not digit blank
            DROP DROP TEMP CONVERT                            answer and re-accept if
            THEN ;                                            is digit convert and
                                                              Leave on stack
: GETNUMBER CLS PROMPT 2 7 Ø ACCEPT ( -- )                    Do the whole thing
            REPRINT ;
GETNUMBER                                                     Autostart on Loading
```

A FURTHER COMMENT ON ACCEPT:
This word will accept up to 5 INTEGERS. It expects on the stack in the followng order:

Number of digits (ND)
Starting Column (col)
Starting Rowu (Row)

3DUP: These parameters are saved for possible future use in blanking out an incorrect input by 3DUP,which duplicates them all,and they remain on the stack beneath everything else happening on top of them until needed. If not needed they are later dropped by DROP DROP DROP.

GOTOXY positions cursor after the PROMPT ready to accept data

ENTER expects a Digit count and deposits characters in the allotted buffer TEMP leaving the count on the stack for use by ?DIGIT

?DIGIT sequentially checks if each of the characters in TEMP are Digits or Non Digits leaving a True (1) if non Digit and a False Flag (Ø) for each digit. These Flags are then added together to produce a single Non zero (non digit) or zero (all digits) flag for IF to use.

IF tests the flags if true proceeds otherwise goes to ELSE.

NOT duplicates the starting parameters for re-use and consumes the top set to Blank out the unacceptable input (non digits) with a blank (Char 32). Using NOT as a name has no bearing as to its action, but it Anglices the definition of accept to make more sense.

MYSELF sends the routine back to the start of itself

ELSE If they are digits the starting parameters are no longer needed so are dropped by DROP DROP DROP.

CONVERT Then converts to a double number the contents of TEMP, leaving it on stack for future use in the program.

NAMING FORTH WORDS.

When naming FORTH definitions a comment I once read continually pops into mind:

> "A beginning programmers description of FORTH will probably include words such as "mind bending", "twisted", or "weird" along with a few undeleted expletives".

I remember annotating in the margin in capital letters VERY TRUE!!!

A further comment was made that FORTH programmes are notoriously difficult to comprehend, even for the programmer. We can probably all remember sitting in front of a BASIC LANGUAGE printout months after the event, scratching our heads thinking " Now why did I do that?"

I believe every effort should be made to have each definitions name describe its intended action. It might waste a few extra bytes to print GETNUMBER instead of GTN, but the time saved later is worth it. The word above GETNUMBER Does exactly what it says(I hope) ie Clears screen(CLS), puts up the prompt(PROMPT), Accepts that data(ACCEPT), And reprints the result(REPRINT).

Obviously not all words within a definition describe clearly the functions they perform such as ?DIGIT, which mainly uses stack and address manipulator definitions. With these definitions an adequate description of their function is vital.

Other words such as NOT may have no bearing on their intrinsic action, but can help the flow of a definition such as used in ACCEPT.

This may seem like an excessively wordy screen to do a simple thing like accept. One could have made the definitions shorter and less readable. Also such basic definitions my be used many times over in the one programme.

Thats all for this month. If anyone has any bright suggestions or questions don't hesitate to ring me on either WK 436861 (as long as I'm not busy) or H 22450 at a reasonable hour (not before work) and I will do my best to help. Next month some string manipulation definitions to accept strings, compare, move , add and find segments !

Richard Terry 30/09/85.

VIEWS FROM FUNNELWEB FARM

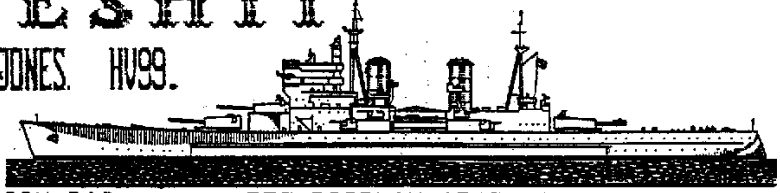
Just a short note this time. FUNLWRITER is now being extended to work as a loader for the E/A Assembler so that Extended Basic by itself can support a complete Assembly programming environment. A new option "Source Edit" loads a TI-Writer Editor, further modified to mimic the E/A editor in its essential features, but retains the superior features of TI-Writer otherwise. We don't use the E/A editor any more. The assembler loader is not finished yet, but is progressing satisfactorily.

WARNING !! The immensely promising Disk Manager 1000 Public Domain program from Ottawa is fatally flawed. It destroys files and disks and not only under the conditions its document warns of. DO NOT USE it on anything you can't afford to trash. It destroyed my FUNLWRITER source code, back-up copy and all. If a good public domain disk manager doesn't show up soon, we are going to have to create one here.

In the last hours before copy deadline I have had a chance to look at Explorer from Miller's Graphics. A truly impressive programming effort, but I'm not sure whether to regard it as game or utility. I find writing programs is THE computer game, and playing other people's is of little interest. The manual gives just a little more guidance than Miller's has already published on TI's shameful secret - GPL. As a utility -- well -- I usually manage to create my own ad hoc means of finding out anything I need to know. If a utility is useful enough to be essential then it is equally important to be able to maintain your own disk backup. I believe that any utility program which does not allow this is so user-hostile as to deserve to be boycotted on principle. Still, as a game with instructional purpose, it's the finest I've yet seen for the TI-99/4a.

"BATTLESHIP"

BY. GARRY JONES. HV99.



```
100 CALL CLEAR :: OPTION BAS
E 1 :: DIM V(18),H(18),F$(9)
,F(9),C(9),VCC(30),HCC(30)::
 RANDOMIZE
110 CALL COLOR(1,9,16,9,4,16
,10,4,16,11,8,16,12,9,16,13,
2,16,14,2,16):: CALL MAGNIFY
(3)
120 FOR A=1 TO 9 :: READ F$(
A),F(A),C(A):: NEXT A :: FOR
 A=1 TO 23 :: READ CHR,A$ ::
 CALL CHAR(CHR,A$):: NEXT A
130 GOSUB 920
140 PTT,PT,TU,FCH,FCR,FCL,FL
 CHITS,YHITS=0 :: FOR A=1 TO
 18 :: V(A),H(A)=0 :: NEXT A
 :: FOR A=1 TO 30 :: VCC(A),
HCC(A)=0 :: NEXT A
150 CALL CLEAR :: CALL SCREE
N(16)
160 FOR A=9 TO 28 :: DISPLAY
 AT(1,A):CHR$(56+A):: NEXT A
170 FOR A=2 TO 21 :: DISPLAY
 AT(A,9):CHR$(128);RPT$(CHR$
(129),19):: DISPLAY AT(A,7)S
IZE(2):USING "##":(A-1):: NE
XT A
180 FOR A=1 TO 21 :: DISPLAY
 AT(A,6)SIZE(1):CHR$(34):: N
EXT A
190 DISPLAY AT(3,1)SIZE(5):R
PT$(CHR$(33),5)
200 DISPLAY AT(9,1)SIZE(5):R
PT$(CHR$(33),5)
210 DISPLAY AT(15,1)SIZE(5):
RPT$(CHR$(33),5)
220 GOSUB 820 :: GOSUB 860 :
: DISPLAY AT(1,1)SIZE(5):"SH
OTS" :: DISPLAY AT(2,1)SIZE(
5):" 0"
230 CALL HCHAR(22,1,32,96)::
 FOR A=1 TO 9
240 DISPLAY AT(22,1):"YOUR "
;F$(A);" POS (";STR$(F(A));"
CHARS)" :: CALL HCHAR(24,25,
32,2)
250 DISPLAY AT(23,2):"HORZ C
O-ORDINATE- " :: ACCEPT AT(2
3,23)VALIDATE(DIGIT):V$ :: I
F V$="" THEN 250 :: V(A)=VAL
(V$)
260 IF V(A)<1 OR V(A)>20 THE
N 250
270 DISPLAY AT(24,2):"VERT C
O-ORDINATE- " :: ACCEPT AT(2
4,23)VALIDATE(UALPHA):H$ ::
IF H$="" THEN 270
280 H(A)=(ASC(H$)-64):: IF H
(A)<1 OR H(A)+F(A)>21 THEN 2
70
290 FOR P=1 TO 9 :: IF P=A T
HEN 300 ELSE IF V(A)=V(P)THE
N 310
300 NEXT P :: GOTO 320
310 IF H(A)>=H(P)AND H(A)<=(
H(P)+F(P)-1)THEN 900
320 FOR B=1 TO F(A):: DISPLA
Y AT(V(A)+1,H(A)+7+B)SIZE(1)
:CHR$((C(A)-1)+B):: NEXT B
330 IF H(A)+F(A)<21 THEN DIS
PLAY AT(V(A)+1,H(A)+F(A)+8)S
IZE(1):CHR$(128)
340 NEXT A
350 CALL HCHAR(22,1,32,96)::
 DISPLAY AT(22,1):"THE COMPU
TER IS LOCATING HIS":"
    OWN SHIPS"
360 FOR A=10 TO 18 :: V(A)=I
NT(RND*20)+1
370 H(A)=INT(RND*20)+1 :: IF
 (H(A)+F(A-9))>21 THEN 370
380 FOR P=1 TO 18 :: IF P=A
THEN 390 ELSE IF V(A)=V(P)TH
EN 400
390 NEXT P :: GOTO 420
400 IF P>9 THEN 410 ELSE IF
H(A)>=H(P)AND(H(A)<=H(P)+F(P
)-1)THEN 370 ELSE 390
410 IF H(A)>=H(P)AND(H(A)<=H
(P)+F(P-9)-1)THEN 370 ELSE 3
90
420 NEXT A
430 CALL HCHAR(22,1,32,96)::
 IF TU=30 THEN 740 :: DISPLA
Y AT(22,1):"YOUR CHOICE  HOR
Z # " :: ACCEPT AT(22,23)VAL
IDATE(DIGIT):VY$
440 IF VY$="" THEN 430 :: VY
=VAL(VY$):: IF VY<1 OR VY>20
 THEN 430
450 DISPLAY AT(23,14):"VERT
# " :: ACCEPT AT(23,23)VALID
ATE(UALPHA):HY$ :: IF HY$=""
 THEN 450
460 HY=(ASC(HY$)-64):: IF HY
<1 OR HY>20 THEN 450 :: TU=T
U+1
```

```
470 CALL VCHAR(VV+1,HH+10,11
2):: DISPLAY AT(1,1)SIZE(5):
"SHOTS" :: DISPLAY AT(2,1)SI
ZE(5):TU :: FOR A=10 TO 18 :
: IF VV=V(A)THEN 570
480 NEXT A :: GOSUB 610
490 IF FCH=0 THEN VC=INT(RND
*20)+1 :: HC=INT(RND*20)+1 :
: FCL=0 :: GOTO 830 ELSE IF
FCH>0 AND FCL>=1 THEN HC=HC-
1 :: GOTO 590
500 IF FCH>0 AND FCR=0 THEN
HC=HC-(FCH+1):: FCL=FCL+1 EL
SE HC=HC+1 :: FCL=0
510 GOTO 590
520 FOR A=1 TO 9 :: IF VC=V(
A)THEN 540
530 NEXT A :: GOSUB 610 :: G
OTO 430
540 IF HC>=H(A)AND HC<=(H(A)
+(F(A)-1))THEN FCH=FCH+1 ::
FCR=FCR+1 :: VV=VC :: HH=HC
:: GOSUB 620 :: PT=F(A)*4/4
:: GOSUB 820 :: GOTO 430
550 IF FCL>0 THEN FCH=0 :: P
T=F(A)*10 :: GOSUB 820 :: GO
SUB 610 ELSE FCR=0 :: GOSUB
610
560 GOTO 430
570 IF HY>=H(A)AND HY<=(H(A)
+(F(A-9)-1))THEN VV=VY :: HH
=HY :: GOSUB 620 :: PTT=F(A-
9)*4/4 :: FYH=FYH+1 :: GOSUB
 850 :: GOTO 490
580 GOSUB 610 :: GOTO 490
590 CALL HCHAR(22,1,32,96)::
 DISPLAY AT(22,1):"COMPUTER'
S CHOICE HORZ#";VC :: DISPLA
Y AT(23,19):"VERT# ";CHR$(HC
+64)
600 FOR DL=1 TO 700 :: NEXT
DL :: VCC(TU)=VC :: HCC(TU)=
HC :: GOTO 520
610 CALL SOUND(50,110,3):: D
ISPLAY AT(24,6):"THAT'S A MI
SS" :: FOR DL=1 TO 200 :: NE
XT DL :: RETURN
620 CALL SOUND(500,-6,3):: C
ALL SPRITE(#1,113,7,VV*8,(HH
+9)*8,#2,117,11,VV*8,(HH+9)*
8):: CALL SOUND(500,-7,3)::
IF A>9 THEN CH=122 ELSE CH=1
21
630 CALL HCHAR(VV+1,HH+10,CH
):: CALL DELSPRITE(ALL):: RE
TURN
640 DATA CARRIER,4,96,SUB #1
,2,100,SUB #2,2,102,SUB #3,2
,100,SUB #4,2,102
650 DATA FRIGATE #1,3,104,FR
IGATE #2,3,107,FRIGATE #3,3,
104,FRIGATE #4,3,107
660 DATA 96,00000000FF3F1F0F
,97,0101070FFFFFFFFF,98,8080
C0C0FFFFFFFF,99,000000000FFF8
F8F0
670 DATA 100,0000000303FFFF7
F,101,808080C0C0FCFEFF,102,0
000000303FFFF7F,103,808080C0
C0FCFEFF
680 DATA 104,0000027E0EFF7F3
F,105,0131113F2AFF55FF,106,1
0101090096FE7CF8
690 DATA 107,0000027E0EFF7F3
F,108,0131113F2AFF55FF,109,1
0101090096FE7CF8,112,FFFFFFFF
FFFFFFFF
700 DATA 121,8142240810244281
,122,FFFFFFFFFFFFFFFF,128,F
F8181818181FF,129,FF0101010
10101FF
710 DATA 113,0000080214021017
211A0D230A000000000000D04490
A840E8C070445
720 DATA 117,28A442A00086000
0D000E000605AA44AA095B201020
1020007000B0082A9A4AA
730 DATA 33,FF,34,0101010101
0101
740 RESTORE 660 :: FOR A=1 T
O 14 :: READ KHR,A$ :: KHR=K
HR+34 :: CALL CHAR(KHR,A$)::
 NEXT A
750 FOR A=10 TO 18 :: FOR B=
1 TO F(A-9):: DISPLAY AT(V(A
)+1,H(A)+B+7)SIZE(1):CHR$((C
(A-9)+33)+B):: NEXT B
760 IF H(A)+F(A-9)<21 THEN D
ISPLAY AT(V(A)+1,H(A)+F(A-9)
+8)SIZE(1):CHR$(128)
770 NEXT A
780 IF CHITS=YHITS THEN L$="
WE JUST TIED" ELSE IF CHITS>
YHITS THEN L$="I WON" ELSE L
$="YOU WON"
790 IF CHITS=0 AND YHITS=0 T
HEN L$="WE'RE BOTH BAD SHOTS
"
800 CALL HCHAR(22,1,32,96)::
 DISPLAY AT(22,6):L$: :"
TRY ANOTHER GAME [Y/N]"
810 CALL KEY(3,KEY,ST):: IF
ST=0 THEN 810 ELSE IF KEY=89
 THEN 140 ELSE END
820 CHITS=CHITS+PT :: DISPLA
Y AT(5,1)SIZE(5):"COMP'S" ::
 DISPLAY AT(6,1)SIZE(5):"SCO
RE" :: DISPLAY AT(7,1)SIZE(5
):CHITS :: RETURN
830 FOR X=1 TO TU :: IF VC=V
CC(X)AND HC=HCC(X)THEN 490
840 NEXT X :: GOTO 590
850 IF FYH=F(A-9)THEN BT=F(A
-9)*10
860 YHITS=YHITS+PTT+BT :: DI
SPLAY AT(10,1)SIZE(5):"YOUR"
 :: DISPLAY AT(11,1)SIZE(5):
"SCORE" :: DISPLAY AT(12,1)S
IZE(5):YHITS
870 IF TU=0 THEN RETURN
880 IF FYH=F(A-9)THEN BT=0 :
: FYH=0
```

```
890 RETURN
900 CALL HCHAR(22,1,32,96)::
 DISPLAY AT(22,4):"YOU JUST
RAN INTO YOUR":"           O
WN SHIP":"!!! TRY ANOTHER PO
SITION !!!"
910 FOR DELAY=1 TO 999 :: NE
XT DELAY :: CALL HCHAR(22,1,
32,96):: GOTO 240
920 CALL MAGNIFY(2)
930 DISPLAY AT(7,7):"**BATTL
ESHIPS**" :: DISPLAY AT(8,13
):"BY" :: DISPLAY AT(9,9):"G
ARY JONES"
940 DISPLAY AT(11,9):"NEWCAS
TLE" :: DISPLAY AT(13,7):"NS
W AUSTRALIA"
950 N=1 :: CL=15 :: R=80 ::
CN=10 :: SP=-20 :: GOSUB 106
0
960 N=5 :: CL=5 :: R=160 ::
CN=210 :: SP=-10 :: GOSUB 10
60
970 N=9 :: CL=4 :: R=96 :: C
N=96 :: SP=-20 :: GOSUB 1060
980 N=13 :: CL=2 :: R=56 ::
CN=1 :: SP=-10 :: GOSUB 1070
990 N=15 :: CL=15 :: R=144 :
: CN=90 :: SP=-10 :: GOSUB 1
070
1000 N=17 :: CL=10 :: R=64 :
: CN=20 :: SP=-20 :: GOSUB 1
070
1010 N=19 :: CL=13 :: R=175
:: CN=200 :: SP=-20 :: GOSUB
 1080
1020 N=22 :: CL=8 :: R=120 :
: CN=96 :: SP=-10 :: GOSUB 1
080
1030 N=25 :: CL=12 :: R=20 :
: CN=90 :: SP=-10 :: GOSUB 1
080
1040 FOR DL=1 TO 999 :: NEXT
 DL
1050 CALL DELSPRITE(ALL):: R
ETURN
1060 CALL SPRITE(#N,96,CL,R,
CN,0,SP,#N+1,97,CL,R,CN+13,0
,SP,#N+2,98,CL,R,CN+25,0,SP,
#N+3,99,CL,R,CN+36,0,SP):: R
ETURN
1070 CALL SPRITE(#N,100,CL,R
,CN,0,SP,#N+1,101,CL,R,CN+13
,0,SP):: RETURN
1080 CALL SPRITE(#N,104,CL,R
,CN,0,SP,#N+1,105,CL,R,CN+13
,0,SP,#N+2,106,CL,R,CN+25,0,
SP):: RETURN
```

MINI WORD PROCESSOR DISSECTED

BY

BRIAN RUTHERFORD

HV99

How did I manage to keep the programme so short I have been asked. Firstly Extended Basic is very powerful with the likes of the ACCEPT AT and DISPLAY AT statements, and best of all the user defined subprogrammes that other makes of computers are only just starting to include in their Basic language's. Secondly a big thank you to Tony Mc.Govern whose extended tutorials on subprogrammes taught me how to use them, which enables me to structure my programmes better.

So let us start looking at how it works, by taking it line by line and branching to the different subprogrammes as they are called.

Ignoring the header and starting at line 160 where the one dimensional array L$ is optioned from zero and dimensioned to 66. The L loop chsnges all the character sets from the cursor set zero through 12 to white foreground with transparant background . Then L is reset to zero as the text line counter, and L$ subscript zero is set to "8000" with the statement L$(0)="8000". The amount of free space available for text.

Line 170 sets the screen to dark blue, then the statement CALL MEN(L$()) calls the subprogramme MEN which is short for menu. Transferring the array L$ to the subprogramme at line 320 SUB MEN(L$()), where the array is known as L$ also. As you can see this subprogramme displays the main menu also the the current string stored in the variable L$(0), the amount of free space available for text. After which control is returned to the main programme and the following

statement. Which is CALL CH(9,K), CH being short for choice. The call statement transfers the value 9 and the variable K to the subprogramme CH, where the 9 is transfered to the variable X, and the variable K is transfered to another variable named K at line 350 SUB CH(X,K). This subprogramme looks after the menu choices, in this instance 9 different choices, later you will see it used with a different number of choices, and each time the number of choices can be transfered to the variable X. You will see that the message "Your choice" is displayed and then a CALL KEY at line 360. A call key 3 is used to set the keyboard to return all keys pressed to upper case, with K being the variable in which the ASCII code for the key that was pressed is placed. If the key pressed is with in the required parameters set by the variable X the subexit is performed else a warning message is displayed line 370 and the programme goes back to line 360 for the user to try again. After a key input has been accepted and the subexit has been performed back to the main programme at line 180, which then subtracts 48 from the value of K, which was transfered from the variable K in the subprogramme CH. Which gives a value between 1 and 9 in this case, for the ON GOSUB statement to opperate on. The programme then branches to lines 200,210,220,230,240,250,260,270,280 as per the value of K-48.

Let us take line 200 first, CALL IN(L$(),L), IN being short for input. As you can see the array L$ and the text line counter L is transfered to SUB IN(L$(),L) at line 460 and the same names are used again. Don't forget all though it is the same array as the array in the main programme and SUB IN knows it by the same name, the variable L is <u>not</u> the same L. A dummy call key is used to reset the keyboard to a full keyboard, that is upper and lower case characters. Then the reminder message on how to exit input mode is displayed on the middle of the screen, and the current string stored in L$(L) is printed on the bottom of the screen. If there is no text in the memory L=0 and L$(0) is where the amount of free space is stored. That is the reason 8000 is displayed on the screen when you first select "Add text". Then line 470 first increments the text line counter L, and then the LINPUT statement waits for text to be entered. You will notice that the input prompt is two quote marks, a null string. That is to stop the computer from displaying a question mark at the start of the input line, also LINPUT was used because it accepts anything that is typed in with no editing. Next line 470 checks to see if the user has typed in the two slashes to exit input mode, if so the text line counter is decremented and the subexit is performed. If the input was not the subexit switch then line 480 checks to see if the array L$ is full or the amount of space left for text is less than zero, that is the value of the string stored in L$(0) is less than zero. If so a warning message is displayed a tone sounded and a subexit is performed, else at line 490 the length of the line of text that was just entered is subtracted from the value of the string stored in L$(0), and returns to line 470 to wait for the next line of text.

I think that is enough for this month, in the meantime for any body who is just starting to come to grips with EXTENDED BASIC and subprogrammes. Why did I use L$(0) to store the amount of text space available, and not a simple numeric variable.
Brian R.

GAME REVIEW "TUKOM KINGDOM"

REVIEW BY

JOHN SMART

"TUKOM'S KINGDOM" is a graphics adventure, written in BASIC, in which you make your way through five screens in search of "THE STONE" (I missed the significance of this).

Monsters hinder your progress by making determined (if unintelligent) attempts to kill you.

The graphics are good, but not brilliant. There are five screens to explore.

You begin in a maze-like dungeon, where you have to get a key to open the door to the next level. A monster (a great ugly hulking black

thing!) will try to stop you, and you have to run it through with your sword. Only one monster appears on the screen at any one time, but a replacement monster appears immediately when you kill one. There is also a "jump" spell in the dungeon, which enables you to teleport to a random position on the screen (very useful for getting out of tricky situations).

The second screen is "THE DARK FOREST", where the monsters move notably faster. This makes for some very hectic action as you struggle to get to the door into the next level.

The third level is "THE LAND OF PITS", where similarly hectic encounters take place.

The forth level is one which I glimpsed only briefly, where the computer told me "YOU ARE THIRSTY".A second or so later a monster permanently cured my thirst!

The fifth (seen only through telling the computer to start at this level) contains the magical "STONE" and, once you get this, your quest is over. When you die (either by being hit by a monster, or by falling down a pit) a gravestone appears in your place, and a screen tells you how many monsters you have killed, and rates your performance ("a poor try" for less than 10 kills, and "a good try" for anything else). One feature I didn't like was the fact that you only get one life; having to start all over again could be quite frustrating. The major fault of the game was the movement of the monsters and of your character; the graphics were slow to update, and this made it difficult to keep a tab on exactly where the monster was. The sound was disappointing; only a few beeps and buzzes, and a short fanfare when you died. The keys seemed a bit sluggish, but this was probably caused by TI's sluggish BASIC as much as by the program itself.

Overall, it is quite a good game, considering that it is written in BASIC. While the slow graphics update is a problem, the graphics are well designed, and the game is quite playable. It gets definitely more exciting at the higher levels, where the monsters move faster and more aggressively. All in all, it is an enjoyable game.

SORTING OUT SORTS

PART 3
BY
JOE WRIGHT
HV99

SORTING OUT SORTS

With only one exception, this Month is devoted entirely to Sorting Routines. Selection Sort, Shell Sort and Quick Sort are included.

THE EXCEPTION.

The screen display used in the Sorts so far discussed have left the screen rather untidy and not easy to read. The Print Line;

```
PRINT S(A)
```

can be removed and the two following lines added in it's place;

```
Z=A-5*INT((A-1)/5)
PRINT TAB(5*Z);S(A);
```

Using this Printing routine will increase the RUN time of the programmes by approx. 12 seconds. Also be aware that the counting of Comparisons and Swops slows the programmes also.

SELECTION SORT.

I have included this Sort because of it's interesting sorting method. Load the original Bubble Sort from tape or disc and then retype the following programme lines.

```
300 FOR OUTER=1 TO 99
310 MINN=S(OUTER)
320 WHERE=OUTER
330 FOR INNER=OUTER+1 TO 100
335 COMPS=COMPS+1
340 IF S(INNER)>=MINN THEN 370
```

```
345 SWOP=SWOP+1
350 MINN=S(INNER)
360 WHERE=INNER
370 NEXT INNER
380 TEMP=S(OUTER)
390 S(OUTER)=S(WHERE)
400 S(WHERE)=TEMP
405 NEXT OUTER
```

HOW IT WORKS!

From what has been noted about the Bubble Sort it might be assumed that this Sort will not be quick because of the two FOR/NEXT loops.
The loop OUTER has two functions;
(1)Selects a Minimum value for testing against all other Data Items on LINE 310.
(2)After each Pass through the INNER loop a new Minimum has been found. This is stored in the Array S at the position pointed to by OUTER on LINE 400.
The loop INNER steps the minimum value in MINN through the Data List. When a value less than MINN is found on LINE 340, that new value is placed in MINN on LINE 350. The Data List location for the new MINN is placed in WHERE on LINE 360. On completion of an INNER loop the minimum value for the Data Items tested has been found. NOTE that the INNER loop starts the Data tests from OUTER+1. This causes the TEST on LINE 340 not to test MINN against itself or any Data Items previously placed in their final Sorted position.
The operation of the Selection Sort can be demonstrated as follows;
Assume that 5 numbers are to be Sorted.
10, 16, 17, 8, 12.

First PASS of OUTER.
OUTER=1;MINN=S(1)=10 WHERE=1

```
INNER=2    INNER=3    INNER=4    INNER=5
  10         10         10         10
  16)=10 NO  16         16         16
  17    SWAP 17)=10 NO  17         17
   8          8    SWAP  8)=10 SWAP 8
  12         12         12  MINN=8 12)=8 NO
                          WHERE=4      SWAP
```

Now place MINN into S(1) and the original value from S(1) into the Array at WHERE ie. Exchange 10 and 8.

Second PASS OF OUTER
OUTER=2;MINN=S(2)=16;WHERE=2

```
INNER=3      INNER=4     INNER=5
   8           8           8
  16          16          16
  17)=16 NO   17          17
  10     SWAP 10)=16 SWAP 10
  12          12  MINN=10 12)=10 NO
                    WHERE=4       SWAP
```

Now place MINN into S(OUTER) ie.S(2) and the value from S(2) into the Array position pointed to by WHERE.
Exchange 10 and 16
Third PASS of OUTER
OUTER=3;MINN=S(3)=17;WHERE=3

```
INNER=4     INNER=5
   8           8
  10          10
  17          17
  16)=17 SWAP 16
  12  MINN=16 12)=16 SWAP
     WHERE S=4    MINN=12
                  WHERE=5
```

Now place MINN into S(OUTER) ie S(3) and the value from S(3) into the Array position pointed to by WHERE.
Exchange 17 and 12.

Fourth PASS of OUTER
OUTER=4;MINN=S(4)=16;WHERE=4

```
INNER=5
8
10
12
16
17)=16 NO SWAP
```

Now place MINN into S(OUTER) ie S(4) and the value from S(4) into the Array position pointed to by WHERE.

The Data List is now Sorted. Run the Sort and record run time, my results are;

2 minutes 20 seconds
4950 comparisons
301 swaps.

That is interesting! Bubble has two FOR/NEXT loops and is like a wet week when running. Have another look at the Comparisons and Swops as compared to these taken by Bubble. Selection Sort reduces both of these quite markedly.

SHELL SORT.

The Shell Sort routine named after it's inventor D.I.SHELL is probably the best known Sort of all the Sorts presented in these notes.
Shell concluded that to accomplish a decreased sort time the number of comparisons made in the sort should be reduced from that made by Bubble sort.
Instead of testing adjacent Data pairs, as in Bubble sort. Shell decided to test Data Items half the Data list length apart on the first PASS of the comparison through the Data list. The distance between the Data Items being tested is the GAP.
On suceeding PASSES the GAP is halved untill GAP=1. The GAP of 1 indicates

the final PASS through the Data list. Note that this final PASS is similar to one PASS of a Bubble Sort.
Using this method Shell substantially reduced the comparisons made by the Sort and as a result substantially reduced the sort run time.
Load the original Bubble Sort and retype the following lines.

```
300 G=100
310 IF G<1 THEN 410
320 G=INT(G/2)
330 N=100-G
340 F=0
350 FOR I=1 TO N
360 P=I+G
370 COMPS=COMPS+1
380 IF S(I)<=S(P) THEN 404
390 SWOP=SWOP+1
400 TEMP=S(I)
```

Now add these lines;

```
401 S(I)=S(P)
402 S(P)=TEMP
403 F=1
404 NEXT I
405 IF F>0 THEN 350
406 GOTO 320
```

RESequence the programme. (RES 100,10). Run the programme and with your trusty Seiko time the programme run time my results were;

RUN TIME = 2 MINUTES
SWOPS = 415
COMP/SONS= 2604

This is a mighty improvement on Bubble Sort which took 4 minutes 16 seconds to Sort the same Data list.
The difference between Bubble Sort and Shellsort RUN times grows with the Data List length. The run time for a Bubble Sort increases in proportion to N squared, while, for the Shell Sort it can be shown that it's run time increases in proportion to N to the power of 1.5.

HOW IT WORKS!

The following is a line by line decritpion of how the programme operates.

```
300 SET G TO DATA LIST LENGTH (100).
310 END SORT WHEN GAP G<1.
320 DETERMINES TEST GAP G .
330 SET LIMIT N=DATA LIST LENGTH-GAP.
340 RESET SWOP FLAG TO 0.
350 FOR/NEXT LOOP I TO N.
360 SET P=I+GAP FOR TEST LINE 380.
370 COUNT COMPARISONS.
380 TESTS DATA ITEMS GAP G APART.
390 COUNTS SWOPS.
400/401/402 DATA SWOP ROUTINE.
403 SETS SWOP FLAG F=1.
404 END OF FOR/NEXT LOOP I.
405 TEST SWOP FLAG IF SET CAUSES A REPEAT OF THE
CURRENT FOR/NEXT LOOP.
406 RETURNS TO LINE 320 FOR NEW VALUE OF GAP G.
```

MORE OF SHELL.

As with ALL things. Including User Groups and Sort Routines. The quest for improvement should never end. In this regard Shell Sort and the H.V.99'ers are not any exception. I have three versions of Shell Sort. The above being the slowest of the three. The following is the fastest of these. All will be available on the H.V.99'ers SORT DISC which will be released through the Club Library when this series of articles are completed. This Sort is known as the METZNER-SHELL SORT. After saving the above Shell reload the original Bubble Sort and then retype the following lines.

```
300 G=100
310 G=INT(G/2)
320 IF G=0 THEN 410
330 K=100-G
340 J=1
350 I=J
360 L=I+G
370 COMPS=COMPS+1
380 IF S(I)<=S(L) THEN 405
390 SWOPS=SWOPS+1
400 TEMP=S(I)
```

Now add these lines,

```
402 S(L)=TEMP
403 I=I-G
404 IF I>1 THEN 360
405 J=J+1
406 IF J<K THEN 350 ELSE 310
```

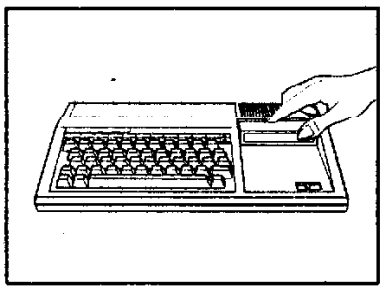
RESequence the programme;
RES 100,10

Run the programme and record the Run Time, Swops, and Comparisons. The results I obtained were;
1 minute 13 seconds
643 Comparisons
238 Swops

This result is mighty quick for poor old T.I. Basic and if the printing time is subtracted that puts this Sort Run time at approx. 45 seconds for 100 numbers.

HOW IT WORKS!
Again so as not to bore experienced programmers and still help our newer programmers I have used a line by line description method to describe how this Sort operates.

```
300 SETS G TO DATA LIST LENGTH
310 DETERMINES TEST 'GAP' G.
320 ENDS SORT WHEN GAP G=0.
330 SETS LIMIT FOR I (LINE 447).
340 RESETS J TO 1 FOR EACH NEW PASS THROUGH LIST.
350 SETS FIRST COMPARISON DATA ITEM NUMBER TO J.
360 SETS SECOND COMPARISON DATA ITEM TO I+G.
   THESE TWO LINES SET THE TWO DATA ITEMS FOR
COMPARISON IN LINE 370 THE 'GAP' G APART.
370 COUNTS COMPARISONS
380 COMPARISON LINE COMPARES DATA ITEMS G APART.
390 COUNT SWOPS
400/401/402 SWOP ROUTINE.
404 WHEN SWOP IS MADE STEPS COMPARISON BACK BY G; TO
TEST FOR FURTHER SWOP.
405 PREVENTS I BECOMING LESS THAN 1.
406 INCREMENTS J+1 FOR NEXT COMPARISON.
407 USES VALUE OF K FOUND ON LINE 330 TO CHECK FOR END
OF THE CURRENT PASS. IF NEW PASS STEPS TO LINE 310. IF
CURRENT PASS TO CONTINUE STEPS TO LINE 350.
```



QUICK SORT

Quick Sort is without any doubt the most complex and quickest sorting technique which will be discussed in these articles. Let me hasten to add that other sorting methods will run more quickly in machine code than Quick Sort in Basic. However the fundamental sorting methods described remains the same irrespective of the programming language.

Load the original Bubble Sort into your computer and retype the following lines.

```
220 DIM S(100),ST(20,2)
300 N=100
310 I1=1
320 J1=N
330 I=I1
340 J=J1
350 STATUS=-1
360 COMPS=COMPS+1
370 IF S(I)<=S(J) THEN 403
380 SWOP=SWOP+1
390 TEMP=S(I)
400 S(I)=S(J)
```

Add these lines.

```
401 S(J)=TEMP
402 STATUS=SGN(-STATUS)
403 IF STATUS=1 THEN 405 ELSE
404 GOTO 408
405 I=I+1
406 GOTO 408
407 J=J-1
408 IF I<J THEN 360
409 IF I+1<J1 THEN 409
```

RES the programme RES 100,10
then add these line.

```
491 P=P+1
492 ST(P,1)=I+1
493 ST(P,2)=J1
494 J1=J-1
495 IF I!<J1 THEN 330
496 IF P=0 THEN 496
497 I1=ST(P,1)
498 J1=ST(P,2)
499 P=P-1
```

RES the programme RES 100,10
then add this line.

```
581 GOTO 330
```

Modify these lines.

```
490 IF I+1)=J1 THEN 530
550 IF P=0 THEN 600
```

Run the programme and record the run time, swops and comparisons.

The results I obtained were;

1 Minute 13 seconds
643 comparisons
238 swops

HOW IT WORKS!

Quick Sort uses the age old technique of taking a large problem and converting it into a series of smaller, simpler problems.

To illustrate the point. Bubble Sort has to make 99 comparisons to sort a Data list 100 long. That is 9801 comparisons in all. If the Data list was divided into 10 list 10 items long the total comparisons now needed to be made is 810.

Quick Sort uses a some what similar technique. You will have noted that the programme has two arrays. The first S(100) is the Data array. The second ST(20,2) is a two dimensional array which is used to store the boundaries of sub files created during the sorting process. The programme has two distinct parts. The first (lines 300-500) is the sort. This section firstly sorts the complete data list about a master item and subsequently sorts each sub file about it's own master item. The second section stores and retrieves sub files boundaries from the stack as equired.

The sort routine uses the first data item of the file being sorted as the master item for that sub file. As the file is sorted Data items are

placed ahead of or after the master item depending on their value relative to the master item.
This divides the data list into two sub file. (Note that the master item is now in it's correct sorted position). The boundaries of the higher sub file as pushed into the stack. The remaining sub file is now sorted. The first data item in this file is the master item for the new sort. When the sort is completed this sub file has been further divided into two sub files. The high sub file has it's boundaries pushed into the stack and the remaining file is sorted.
This process of dividing the file continues, until the remaining file contains only on one data item. At this point a file boundary is popped from the stack. The sorting process continues untill the file has only one data item at which point another set of boundaries of popped from the stack. This file is then sorted. This continues untill the stack is empty at which point the data list has been completely sorted.
The Array ST(20,2) is the stack array in which the sub file boundaries are stored (pushed). The stack operation is last in first out. This operation is achieved by incrementing or decrementing the value in the varaible P on lines 491 and 499. P increases when a set of boundaries is pushed into the stack. When a set of boundaries is popped from the stack P is decreased. When P=0 the stack is empty. If P=0 and no sub file exists between I1 and J1 then the sort is comleted.

LINE BY LINE DESCRIPTION.

220 SET UP DATA LIST AND STACK ARRAYS.
300 SET N= DATA LIST LENGTH.
310 SET BOUNDARY FOR FIRST PASS THROUGH LIST.
320 SET BOUNDARY FOR FIRST PASS THROUGH LIST.
330 LOAD BOUNDARY INTO VARIABLE FOR SORT
340 LOAD BOUNDARY INTO VARIABLE FOR SORT.
350 SET STATUS TO INDICATE MASTER ITEM AT LOCATION POINTED TO BY I. IF STATUS -1 MASTER ITEM AT I.
IF STATUS 1 MASTER ITEM AT J.
360 COUNT COMPARISONS
370 DATA COMPARISON.
380 COUNT SWOPS.
390/400/401 DATA SWOP ROUTINE.
402 REVERSE STATUS SIGN AFTER SWOP.
403 IS MASTER ITEM AT I OR J
404 UNCONDITIONAL BRANCH AROUND COUNT.
405 INCREASE I BECAUSE MASTER ITEM UNDER J.
406 UNCONDITIONAL BRANCH
407 DECREASE J BECAUSE MASTER ITEM UNDER I.
408 IF STILL DATA ITEM TO TEST RETURN AND CONTINUE UNTILL I=J.
409 IF I+1)=J1 THEN NO FILE TO BE PUSHED INTO STACK. ADVANCE TO POPPED VALUE FROM STACK.
491 INCREASED STACK COUNTER BY 1.
492 PUSH I+1 INTO STACK.
493 PUSH J1 INTO STACK
494 NEW UPPER BOUNDARY FOR SUB FILE.
495 IF SUB FILE EXISTS THEN RETURN AND SORT.
496 IF STACK IS EMPTY END SORT.
497 POP BOUNDARY FROM STACK.
498 POP BOUNDARY FROM STACK.
499 DECREASE STACK COUNT BY 1.
581 RETURN AND SORT NEW SUB FILE POPPED FROM STACK.

CONCLUSIONS.

Next month will be a wrap up of the Basic Sort Routines. The Sorts discussed this month represent chalk and cheese. The first, Selection Sort is a useful sort for small Data lists. As the data list lengthens so the Run time increases quickly. For long Data lists something else is needed. Shell Sort presents some savings in Run time over Bubble but again, the Run time starts to become excessive for longer Data Lists. The Metzner-Shell Sort is ideal for Data List from 50 through to a couple of Hundred. The main advantage of Shell is that it's operation is relatively simple. Shell also requires less Memory space than Quick sort. Repeating a comment from the first article, "a simple programme which performs the required task and requires less memory; beats hands down a complex programme".
The outstanding feature of Quisk Sort will be illustrated next month when a full analysis is done of all the Sorts over a range of Data list lengths. The Run times presented next month will not include the time for printing routines and counting of Swops and Comps.

ANY MORE?.

If you read this then it is obvious that you that you have read the article. If you have gained the impression that I am interested in Sorting Routines you are right! I have a Library of Sorting Routine which range from brilliant to awful. In between these are good, different, unusual, crazy, interesting etc. If you happen to find a Sort Routine which has not been included in these article and also looks a bit different I would be interested in obtaining it. If you are after a Sort Routine then also please don't hesitate to contact me. I could have just what you are looking for.
A.W.

LIBRARY NEWS BY ALAN LAWRENCE

HI 99'ers,
****Adventurers****

Where to start! Well as in all good adventures the begining seems the best place. I have 3 game souls ready to stride forth! We will start at the October meeting and see how far we journey each month hence, if the way is long and dangerous, we may have to call for large parent type guardians to accompany us and try WEEKLY meets to speed us further on our choosen path. Any more volunteers will be welcome.

Disk news

Anyone read the pros and cons of flipping floppies or doubling your disk capacity cheaply??? Save $$$$ 64 thousand, for a small outlay of $US 9.95. One of our members did and received a cardboard replica of a DISK and a shaped insert. The idea being to place the disk in the jig , mark inner light hole outer write hole positions then place insert through centre between case and disk to protect the surface while punching the hole. Don't waste your money. Steve Taylor had an old disk case and used THE front as a template to mark the holes and a piece of the back as the insert. I used half a case and combined both as shown in above sketch for the club library and it wil be demonstrated and is available for use by any person interested in taking risks!!

Note Sectors have been lost as the coating removed from surface during normal rotation is deposited when disk direction is reversed. As the EXPERTS are in number so are the opinions for and against.

----PUBLIC DOMAIN----

As our policy is of FREE EXCHANGE of ideas, software, articles ect. We of the HUNTER VALLEY 99'ers have with the generosity of talented members donated 2 Disks and several programs to the PUBLIC DOMAIN. The first being FUNELWRITER out of the FUNNELWEB FARM stable. And the second a Disk cataloguer that lists 3 cols, numbers the progs and gives a disk report. This IS from el Presidente.

As well we have the Cassette Mini Wordprocessor by B.Rutherford. Gary Jones on Games and Music. and as well Basic classes each week. A Full listing of all PUBLIC DOMAIN software with Credits of origin is elsewere in this Newsletter. It is our aim to expand ours and your club library by Free Exchange. Already we have had interchange of DISKS from the U.K.(TI*MES,STAINLESS SOFTWARE) Australia.(C.H.U.G.,A.T.I.C.C. Tas.T.I.C.U.G.)As well as technical help and co-operation from T.I.U.P. Thanks to all clubs who sent disks to us First or in Exchange.
Also the person for heaps of tapes

Have you heard about AMNION HELPLINE they run a FREE ACCESS LIBRARY of a large collection of PUBLIC DOMAIN software(about 2700 pragrams)Their minimal copying fee for whole volume programs is reasonable and they also have a collection of FREEWARE programs in circulation. The Address AMNION Free Access Library. 116 Carl Street. San Francisco CA 94117 U.S.A. was passed on to me by Stephen Shaw of STAINLESS SOFTWARE. 10 Alstone Road, STOCKPORT, Cheshire.SK4 5AH. U.K.

Library Access

Any clubs or indivuals interested in obtaining any PUBLIC DOMAIN software in volume disks have 2 choices
(a) Send blank initialised disks to us with return postage or send us disks with programs on it and we will send at our cost an equal number of disks filled with programs requested or volume disks.
(b) We can supply programs,or volume disks on our disks for the cost of disk and PP($4.00)

We are not a SHOP and do not sell for profit, any excess is used to buy more material for club usage.
Cassette tapes only to be available at club monthly meetings owing to the problems associated with them.
Your library has about 1000 programs on 80 disks.(and hopes for further exchanges to arrive)full listings to be available at meetings now. A number of first class commercial assembly programs and modules have been bought by members and form part of the clubs demo library. Also we are starting a module library which will be run by Bob.MacClure. Details to be announced at meet in HV'99'er NEWS

HV99'er

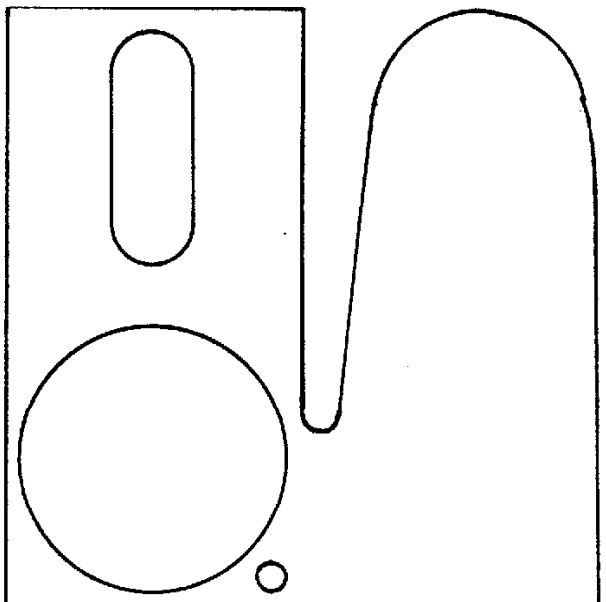
Thats it for now,
Happy programming
Al Lawrence.

PUBLIC DOMAIN DISKS

#	TITLE	SOURCE	COMMENT
001	FUNLWRITER	HV99'ers	T.Mc G.
002	JW-DISKCAT	HV99'ers	J.W.
003	HV99-PDL01	HV99'ers	Various
004	TIWR+MPLAN	TEXAS-INST.	UGPD'83
005	SSPD1	STEPHEN SHAW	UGPD'85
006	DM-1000	OTTAWA 99.UG	B.Caron
007	NO-FRILLS	PD'85(TIshug)	B.Cabot
008	NEATLISTER	FREEWARE	Michael
009	FREEDUMP	FREEWARE	Michael
010	ALROUTINE1	NEW-HORIZONS	Clulow
011	ALROUTINE2	NEW-HORIZONS	Clulow
012	PROGRAMS	CHANNEL 99	Various
013	FORTH		XB load
014	FORTH		MM load
015	FORTHSCODE	TEXAS-INST	2-DISKS
016	FORTH	TEXAS-INST	UGPD'83
017	FORTH-DEMO	AMNION Free Access	
018	A*38	AMNION "	Games
019	A*39	AMNION "	Games
020	E* 8	AMNION "	Bus/Fin
021	F* 7	AMNION "	TecMath
022	G* 6	AMHION "	PerHome
023			
024			
025	TO BE FILLED AS WE EXCHANGE ??		
026			
027	PERHAPS YOU CAN HELP US ?		
028			
029	HELP YOUR GROUP ????		
030			

AVAILABLE NOW FROM
HV99

WHY BUY AN IMITATION



CUT TEMPLATE FROM AN OLD OR DAMAGED FLOPPY. USE AS A GUIDE FOR MARKING INDEX AND WRITE PROTECT HOLES. INSERT FINGER BETWEEN OUTER CASING AND DISK TO PREVENT DAMAGE WHEN THE HOLES ARE BEING PUNCHED. ENSURE PROTECTIVE SIDE OF THE TEMPLATE ONLY TOUCHES THE DISK

THE ORIGINAL HV99
FLOPPY FLIPPER FINGER

FILE NAME :DSK2.PUBLIC-DOM DATE-20/09/85
TOTAL NUMBER OF ENTRIES= 39

1 ADDFILE.....JW-DISKCAT	14 FORMA2.....FUNLWRITER	27 MPBASE.....TIWR+MPLAN
2 CHARA1.....FUNLWRITER	15 FORMA2.....TIWR+MPLAN	28 MPCHAR.....TIWR+MPLAN
3 CHARA1.....TIWR+MPLAN	16 FORMA4800A....TIWR+MPLAN	29 MPDATA.....TIWR+MPLAN
4 DELETE.....JW-DISKCAT	17 FORMA4800B....TIWR+MPLAN	30 MPINTR.....TIWR+MPLAN
5 DM1000/1.....DM-1000	18 FORMATDOC.....TIWR+MPLAN	31 NEWFILE.....JW-DISKCAT
6 DM1000/2.....DM-1000	19 FUNNELDOC.....FUNLWRITER	32 OVERLAY.....TIWR+MPLAN
7 DPC.....FUNLWRITER	20 LOAD.....DM-1000	33 PRACTICE.....TIWR+MPLAN
8 EDITA1.....FUNLWRITER	21 LOAD.....FUNLWRITER	34 PRACTICE1.....TIWR+MPLAN
9 EDITA1.....TIWR+MPLAN	22 LOAD.....JW-DISKCAT	35 PRINT-DV80....DM-1000
10 EDITA2.....FUNLWRITER	23 LOADER.....DM-1000	36 PRINTOUTTR....JW-DISKCAT
11 EDITA2.....TIWR+MPLAN	24 MENU.....JW-DISKCAT	37 SORTA.....JW-DISKCAT
12 FORMA1.....FUNLWRITER	25 MGR1.....DM-1000	38 SSORT.....JW-DISKCAT
13 FORMA1.....TIWR+MPLAN	26 MGR2.....DM-1000	39 END

DISK REPORT
****

DISKNAME	AVAILABLE	USED
DM-1000	117	241
FUNLWRITER	166	192
JW-DISKCAT	250	108
TIWR+MPLAN	27	331

SAMPLE PRINTOUT OF J.W-DISCAT
20 DISKS OR 360 PROGRAMS
ASSEMBLY LANG. SORT ROUTINE

CLASSIFIEDS.

FOR SALE: JIM THREADGATE has a complete TI system for sale including console, PEB with RS 232, 32K, disk drive and disk controller card plus a large assortment of modules and software. If you are interested give Jim a call on 335610.

WANTED: Tony MC GOVERN is after a stand alone disk controller unit. If you have one or know of someone who has one for sale you can contact Tony at 523162.

FOR SALE: "Mint" second hand fully equipped expansion box with 32K. RS 232. Double Density disc controller and twin DD DS. Chinon Drives (professionally enhanced power supply included) at a very "keen" (negotiable) price!!! (We'd like to see it go to a good home...) Bernie Elsner C/O TIUP.

FOR SALE: ALAN LAWRENCE has his NAVARONE DISK FIXER MODULE up for sale. If you would like to be able to take a look inside your software I'm sure Al would be more than happy to sell it to you. contact 486509.

FOR SALE: Two "EXTERNAL" disk drives. The first is a genuine brand new,unused TEXAS INSTRUMENTS drive which comes complete with its own power supply and cables. You can buy the same drive from IMAGIC for $400 but I'll sell you mine for $200. The second "EXTERNAL" drive is a near new "mint" MPI, only a few months old which has had very little use. A piece of cake to hook up to your PEB, you can have it for $150. If you are interested in either (or both) of these drives contact Steve on 487076.

WANTED: MODULES for the module library. If you have any old, rarely used modules why not lend them to the module library so that others might get some enjoyment from them. These modules will always remain your property and you can reclaim them any time that you like. Bob Maclure has taken on the task of Module Librarian and I'm sure he will be more than grateful for any donations or help you can give. Bob will outline the way that the library will be run at the next monthly meeting.

See you all at the Monthly Meeting. STEVE TAYLOR.

PS. Don't forget we still require material for publication in the magazine.