Aug 85

HUNTER VALLEY 99'ERS NEWS

TI 99/4A

HOME COMPUTER NEWSLETTER

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 1985

A.WRIGHT	PRES.	PH.468120
P.COXON	SECT.	PH.751930
B.RUTHERFORD	TRES.	PH.490184
A.LAWRENCE	LIBR.	PH.486509
S.TAYLOR	EDIT.	PH.487076
A.BYRNE	TECH.	PH.490520
T.MCGOVERN	TECH.	PH.523162
B.MAC.CLURE		PH.437431
B.WOOD		PH.662307
D.WINTON		PH.591882

NOTICE BOARD

Good News, the reported sale of the old Council Chambers has apparently fallen through at the moment. Hopefully this will mean that we should be able to continue to meet there for quite a while to come.
Speaking of meetings, don't forget the weekly basic classes held by Garry Jones and Joe wright each Tuesday night at 7 PM. Also our regular monthly meetings are proving to be extremely popular with large numbers of members and their families attending.
There are some great software demonstrations being organised for future meetings including PLATO and our very own Australian produced software GRAPHX. Who knows I may even be able to talk my wife into giving another talk on LOGO, I certainly hope so as I missed her last talk.
Don't forget the dates of future montly meetings, put a ring around the dates on the calender now. 13/8/85 and 10/9/85.

On the subject of meetings, Tony MC.Govern has his ASSEMBLER classes on the last Tuesday of each month and our enthusiastic Forth Group meet on most Thursday nights and from what I've heard just about every other night of the week as well!!!!
As my Sister-in-Law is the Co-ordinator of the Newcastle MICROBEE Users-Group, tentative plans have made to organise an exchange invitation where members of both groups will be invited to attend montly meetings. This will give both clubs the opportunity to show their machines capabilities as well as exchange information and ideas. Anyone wishing to attend one of the Microbee meetings please let us know.

****
* SECRETARY'S NOTES *
****

THE FIRST THING I WOULD LIKE TO DO IS THANK AL AND DAVID FOR THEIR DEMONSTRATIONS AT OUR LAST MEETING, IT WAS A HUGE SUCCESS, KEEP UP THE GOOD WORK BOYS. VERY LITTLE MAIL THIS MONTH, BUT WE ARE STARTING TO RECIEVE NEWS LETTERS FROM OTHER GROUPS AND THESE ARE AVAILIBLE TO EVERY ONE FROM THE LIBRARIAN. NOW SOME INTERESTING NEWS; WE ARE TESTING OUT A MODEM THAT HAS COME TO OUR NOTICE. A 300/1200/75, AUTO DIAL, PHONE INCLUDED, FOR THE MERE COST OF $189; IT IS MADE RIGHT HERE ON THE CENTRAL COAST BY MICROBEE, WE HOPE TO HAVE ALL THE RELATIVE INFORMATION BY THE NEXT MEETING; DETAILS PUBLISHED IN THE NEXT NEWS LETTER. NO DOUBT YOU HAVE A DSE. OR BIG W STORE NEAR BY ,YOU WOULD RATHER GO IN DSE AND THE LITTLE LADY INTO BIG W. JEFF DANIELS A WISE SHOPPER INFORMS US DSE. HAS TWO $99 SPECIALS, A PRINTER/PLOTTER AND A DOT MATRIX PRINTER; ON ONE, THE SIZE OF PAPER USED IS LIKE ON A CALCULATOR THE OTHER IS FIVE 1/2 INCHS WIDE, SO WHAT YOU SAY! WELL YOUR WIFE HAS BEEN HIDING THE FACT THAT BIG W HAS A ELECTRIC TYPEWRITER/PRINTER/PLOTTER WITH A PARALLEL INTERFACE, FOUR COLOURS OR THREE AND ERASE PEN. IT WILL DO PIE, LINE AND BAR GRAPHS, IT HAS 1 LINE MEMORY, THREE PRINT SIZES, 40, 80, 160 CHARACTERS AND PRINTS ON 8.7 INCH PAPER (A4). NOW COMES THE PRICE $289. NOW THEN, IF THE WIFE WANTS TO PRACTICE HER TYPING SKILLS BUY HER ONE THEN SHE MIGHT LEND IT TO YOU IF YOU ASK HER KINDLY. CHRISTMAS IS LOOMING UP AT A FAST RATE AND I WOULD LIKE TO REMIND YOU ALL THAT WE ARE LOOKING FOR TIPS, HINTS AND PROGRAMES FOR THE BUMPER EDITION SO IF YOU WOULD LIKE YOUR NAME UP IN LIGHTS LETS HEAR FROM YOU. RUNNING OUT OF MEMORY BUT BEFORE I GO I MUST MENTION OUR CLASSES THAT ARE RUNNING AT THE MOMENT AND YOU MIGHT LIKE TO COME TOO. THESE ARE THE CLASSES AND PEOPLE TO CONTACT. BEGINERS BASIC=JOE ON 468120; ADVANCED BASIC=GARY ON 573744EX BASIC AT MAIN MEETINGS, IF YOU WANT A CERTAIN TOPIC ON XB RING TONY SO HE CAN PLAN AHEAD ON 523162; FORTH GROUP ON FORTNIGHTLY ON THURSDAYS RING RICHARD TERRY ON 436511 . I MUST CONGRATULATE BRIAN RUTHERFORD FOR HIS MINI WORD PROCESSOR I FIND IT IS VERY EASY TO USE AND IT WILL BE A GOD SEND TO OUR EDITOR SO HE WON'T HAVE TO READ MY HAND WRITING. PETER C.

Editorial

Welcome to ISSUE #3 of HV 99'ERS NEWS; I'm sure that you will find some quite interesting reading this month as we have several new contributors as well as some really good programmes for you to key in.

Brian Rutherford presents Part 2 of his Mini Word Processor program, MINI-FORMATTER. Now all your letter writing will be straight forward and professional looking; you may even be tempted to produce a small article for inclusion in the magazine!!. Brians next contribution for the magazine will be a complete autopsy of the programme with all the how's, why's and wherefore's.

El'PRESIDENTE continues with his tutorial on Sorting routines. He showed me an example of one of the programmes running the other day; a bubble sort routine that was so self explanatory that even I understood what was going on! Forth programmers take heart, Joe intends producing some Forth sorting routines in future issues.

On the subject of FORTH Richard Terry treats us to a description of how he has managed to "struggle forth" and shares with us some of the tricks he has picked up along the way. Thanks a lot Richard but please dont stop now as we would like to hear more from you in the future. Perhaps in the meantime other members of the FORTH group may be willing to pass on some of their hard won knowledge.

Thanks to Geoff Daniels of Rathmines we have our first letter to the Editor. Geoff has also included for publication a program called MULTI-WAY-MATHS, which if you have school age children is well worth keying in. The programme is quite flexible and Geoff includes instructions for adjusting the level of difficulty.

Tony MC.Govern returns with another installment of EXTENDED TUTORIAL, further explaining ways and means of compressing program length. I know there are a lot of people out there who can hardly wait to get there hands on each new installment. Thanks Tony

This month we also have another new contributor, John Smart from Maitland. John has kindly donated to this issue a copy of one of his very successful commercial programmes "STARGUNNER", including full instructions on how to play the game. We look forward to meeting you at one of our monthly meetings, John Perhaps you may be able to demonstrate some of your other programmes?

Continuing with his prolific contributions to the magazine Tony MC.Govern in collaboration with his son Will has once again produced two articles for the magazine. One of them, FUNLWRITE, is something special. FUNLWRITER, guaranteed to be the best word processor programme available forthe TI. And best of all,in true user group spirit, Tony has made the programme PUBLIC DOMAIN. I'm sure this programme will generate a lot of interest amongst user groups world wide, not to mention people who have been selling word processor programmes at greatly inflated prices!!

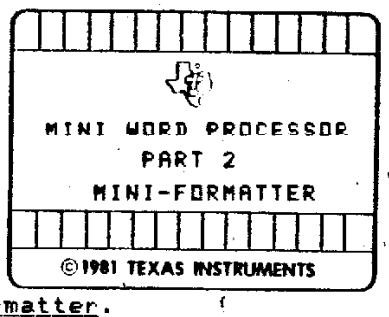
Anyone interested in playing around with prime numbers will enjoy Craig MAC CLURE's programme "Prime Number Generator". Thanks Craig.

Last but not least our hard working Librarian Al. Lawrence returns with more helpful tips on getting your tape recorder sorted out to enable successful loading of club software, as well as a book review and more.

Also this month are a few hints and tips which although are not new to the more experienced programmers may help some of the newer members of our group.

Still haven't received any scores for the SOFTWARE HALL OF FAME or any entries for the club Logo competition !!

I recently had the opportunity to spend a month in Tasmania (working unfortunately). Whilst there I managed to contact Leon Lonegran the Secretary Treasurer of the Tassie TI. Group. Since then our goups have exchanged newsletters and software , we look forward to close contact between the two groups in the future.



Mini Formatter.

Well as you can see I managed to get the formatter programme for Mini Word Processor up and running for this issue. Not only does it convert the cassette files into files that TI Writer can read, it will output directly to a printer also.

I hope you wrote the number of lines that were saved on the cassette, as that is the first thing you will be prompted for when you run the programme. Then after the data is loaded, the screen shows you two choices, 1 Output 2 Exit. Press 1 for output, and the screen changes to look like this!-

Output to
1 Printer
2 Disk
3 Both

If you press 1, PIO is displayed opposite the word Printer as the default. After accepting the default or typing in your printers device file name, the screen clears again. You are then prompted for a line length 10 to 80 characters long. If you input a value outside those parameters the cursor just goes back for you to try again. Next you are prompted for a left margin, if the line length plus the left margin exceed 80, a warning message is shown on the screen, just press any key and the programme goes back to let you have another go at inputing a line length and left margin. The next promp is for page size A4 or quarto, again you only need to press 1 or 2. The programme still prints the same number of lines to a page, only A4 has a larger margin a top and bottom of the page. Then the last prompt is Right justified Y/N. Press Y or N, upper or lower case is does not matter, the screen clears again and the message "Working on it" is displayed. When the printout is finished the programme goes back to the first screen "1 Output 2 Exit" .

When you select Disk output, the

default DSK1.FILE is displayed opposite the disk option, press enter to accept the default or change as necessary, again you will be prompted for a line length the same as for the printer output. But that is all, the file is then printed straight to disk in DISPLAY VARIABLE 80 formatt. When that is finished, you are taken back to the output or exit screen again.

The third option of course takes you through both sets of prompts and does both jobs together, But dont try that if you have done a CALL FILES(1) earlier, as it is printing to 2 files at once.

When you type your text in with the word processor, the following commands can now be used.

To tab a line over a number of places type //number of places/ and then the line. ie.

```
//49/J.  Blow
//50/35 Whatsit St.
//51/Kings Cross.
```

Pressing enter after each line. In that way you can tab your address over to the right side of the page with out having to count the spaces. When using that command in that type of way, due regard for the line length you are going to specify must be taken into account. If the number of spaces you tab over and the length of the text exceed the line length you specify in the formatter programme, the line will be wrapped around into the next line along with many other sorts of funnies. If you were only using the command to indend a line, then the full 5 lines can be typed in. Lastly to centre a line or a heading type /c/ or /C/ and then the line or heading, will centre it. Again take care with the line lengths.

I think those two little commands will make life a lot easier, especialy letter writting, also the tabs are added to the lines befor they are printed to disk as well as the printer. And dont forget the editor is waiting for your letters and articles to start rolling in dont let him down.

Brian R.

```
100 REM !~~~~~~~~~!
110 REM !  MINI   !
120 REM !FORMATTER!
130 REM !   BY    !
140 REM !BRIAN  R.!
150 REM !_________!
160 OPTION BASE 1 :: DIM L$(
66):: DISPLAY ERASE ALL :: F
OR L=0 TO 12 :: CALL COLOR(L
,16,1):: NEXT L :: CALL SCRE
EN(5):: L=0
170 CALL RF(L$(),L)
180 DISPLAY AT(10,8)ERASE AL
L BEEP:"1 Output": :"
2 Exit" :: CALL CH(2,K):: IF
 K=50 THEN DISPLAY ERASE ALL
 :: STOP
190 CALL CUTUP(L$(),L,K):: I
F K=49 OR K=51 THEN CLOSE #1
 :: IF K=49 THEN 180
200 CLOSE #2 :: GOTO 180
210 SUB KC :: DISPLAY AT(24,
1)BEEP:"Press any key to con
tinue"
220 CALL KEY(3,K,S):: IF S<1
 THEN 220
230 SUBEND
240 SUB CH(X,K):: DISPLAY AT
(23,4)BEEP:"Your choice"
250 CALL KEY(3,K,S):: IF S<1
 THEN 250 ELSE IF K>48 AND K
<(X+49)THEN SUBEXIT
260 DISPLAY AT(24,1)BEEP:"A
number between 1 and";X :: G
OTO 250
270 SUBEND
280 SUB RF(L$(),L)
290 DISPLAY AT(24,1)ERASE AL
L BEEP:"How many lines to re
ad" :: ACCEPT AT(24,24)VALID
ATE(DIGIT):A$ :: IF A$="" TH
EN 290 ELSE L=VAL(A$)
300 OPEN #1:"CS1",INTERNAL,I
NPUT ,FIXED 192 :: FOR I=1 T
O L :: DISPLAY AT(12,4)ERASE
 ALL:"Reading line number";I
 :: INPUT #1:L$(I):: NEXT I
310 CLOSE #1 :: SUBEND
320 SUB CUTUP(L$(),L,K):: CA
LL LO(K):: P$="" :: I=0 :: J
=1
330 PL=I :: P$="" :: FOR I=I
 TO L :: LL$=P$&L$(I):: P$="
" :: IF LL$="" THEN CALL PLI
NE(LL$,K,0):: GOTO 430
340 LL=LEN(LL$):: IF SEG$(LL
$,J,3)="/C/" OR SEG$(LL$,J,3
)="/c/" THEN LL$=RPT$(" ",IN
T(PL-LL)/2)&SEG$(LL$,4,LL):: 
 CALL PLINE(LL$,K,0):: GOTO
430
350 IF SEG$(LL$,1,2)="//" TH
EN P=POS(LL$,"/",3):: LL$=RP
T$(" ",VAL(SEG$(LL$,3,P-3)))
&SEG$(LL$,P+1,LL):: LL=LEN(L
L$)
360 IF LL<=PL THEN CALL PLIN
E(LL$,K,0):: GOTO 430
370 FOR J=PL+1 TO I STEP -1
:: P$=SEG$(LL$,J,1):: IF P$<
>" " AND P$<>"." THEN 420 ELS
E P$=SEG$(LL$,1,J-1):: LL$=S
EG$(LL$,J+1,LL):: LL=LEN(LL$
)
380 IF LL$<>"" THEN CALL PLI
NE(P$,K,1):: GOTO 370
390 IF I=L THEN CALL PLINE(P
$,K,0):: SUBEXIT
400 IF SEG$(L$(I+1),1,3)<>"/
C/" AND SEG$(L$(I+1),1,3)<>"
/c/" AND SEG$(L$(I+1),1,2)<>
"//" AND SEG$(L$(I+1),1,4)<>
"    " AND L$(I+1)<>"" THEN
430
410 CALL PLINE(P$,K,0):: P$=
"" :: GOTO 430
420 NEXT J
430 NEXT I :: SUBEND
440 SUB LO(K):: DISPLAY AT(6
,1)ERASE ALL BEEP:"Output to
" :: DISPLAY AT(8,6):"1 Prin
ter": :"     2 Disk": :"
 3 Both" :: CALL CH(3,K)
450 IF K=50 THEN 470
460 DISPLAY AT(8,18)BEEP:"PI
D" :: ACCEPT AT(8,18)SIZE(-3
):P$ :: IF P$="" THEN 460 EL
SE OPEN #1:P$ :: IF K=49 THE
N SUBEXIT
470 DISPLAY AT(10,18)BEEP:"D
SK1.FILE" :: ACCEPT AT(10,18
)SIZE(-9):P$ :: IF P$="" THE
N 470 ELSE OPEN #2:P$,OUTPUT
,DISPLAY ,VARIABLE 80
480 SUBEND
490 SUB PLINE(P$,K,S):: IF K
 THEN 570
500 DISPLAY AT(4,4)ERASE ALL
 BEEP:"Line length 10-80": :
"   Left margin"
510 ACCEPT AT(4,24)VALIDATE(
DIGIT):P$ :: IF P$="" THEN 5
10 ELSE LL=VAL(P$):: IF LL<1
0 OR LL>80 THEN 510 ELSE IF
S THEN 560
520 ACCEPT AT(6,24)VALIDATE(
DIGIT):P$ :: IF P$="" THEN 5
20 ELSE LM=VAL(P$)
530 IF LL+LM>80 THEN DISPLAY
 AT(11,1)ERASE ALL BEEP:"LIN
E LENGTH PLUS LEFT MARGIN IS
 LONGER THAN EIGHTY..." :: C
ALL KC :: GOTO 500
540 DISPLAY AT(8,8):"Paper s
ize" :: DISPLAY AT(10,4):"1
A4": :"   2 Quarto" :: CALL
CH(2,X):: IF X=49 THEN X=11
ELSE X=9
550 DISPLAY AT(23,2):"Right
justified Y/N" :: CALL KEY(3
,J,S):: IF J<>78 AND J<>89 T
HEN 550
560 DISPLAY AT(12,10)ERASE A
LL:"Working on it" :: L=0 ::
 K=LL :: SUBEXIT
570 IF K<>49 THEN PRINT #2:P
$ :: IF K=50 THEN SUBEXIT
580 IF J=89 AND S=1 THEN CAL
L BJ(P$,LL)
590 IF L=0 THEN PRINT #1:""
:: PRINT #1:""
600 PRINT #1:TAB(LM);P$ :: L
=L+1 :: IF L<56 THEN SUBEXIT
610 FOR I=1 TO X :: PRINT #1
:"" :: NEXT I :: L=0 :: GOTO
 KD
620 SUB BJ(P$,LL):: P=0 :: I
F LEN(P$)=LL THEN SUBEXIT
630 A=2 :: FOR J=1 TO LL-LEN
(P$)
640 P=POS(P$," ",P+A):: IF P
=0 THEN A=A+1 :: GOTO 640
650 P$=SEG$(P$,1,P)&CHR$(32)
&SEG$(P$,P+1,LEN(P$)):: NEXT
 J :: SUBEND
```

SORTING OUT SORTS

As promised last month we will now take a look at Sorting String Variables. Firstly, change the Bubble Sort in the previous Article to a String Sort. Add '$' sign to the Array S(100) so it becomes S$( ) on Lines; 230,250,260,340,360,370,380 and 430 Add '$' to TEMP on lines 360 and 380 This has changed all the Numeric Variables to String Variables.
The significance of this will become apparent during this discussion. Run the programme and inspect the resultant "Sorted" Data List. It is not in Numerical order! There is SOME kind of order but what has happened?.The secret to this occurance lies in what happens when when a number is entered into a String variable.

String$ Numbers.

When a number is entered into a String variable, BASIC no longer interprets it as a number.It has lost it's Numeric identity and has becomes a character. Refer to the USERS REFERENCE GUIDE page III-1 for the ASCII Character Codes available at the T.I.99/4A Keyboard. The character is now identified by Basic, only by it's ASCII Code number. The shape of the character or any numerical value formerly associated with the character has no significance to Basic when handling a String variable. As a result of this during a String Sort a number is treated no differently to any other character; a,A,z,Z,+,(for example.

String Sort.

Changing the Bubble Sort from a Numeric Sort to a String Sort changed it to a Sort which uses the ASCII Code to re-arrange the Data List into,the prefined order. Run the String Bubble Sort again and take particular note of the Sorted Order of 88,881,882 etc. The String Sort compares the Data Items character by character. In the case of 88 and 881 the first two characters are found to have the same ASCII Code Number. The third characters are then compared. A null (ASCII 32) is found in the third character position in 88. The third character in 880 is ASCII Code 48 (zero). This results in 88 being placed ahead of 880 in the Sorted Data list.

Alter the following programme lines in the String Bubble Sort to that shown below,
Line - 240 FOR A = 1 TO 10
Line - 280 N=10
Line - 380 FOR FIRST = 1 TO 9
Line - 420 FOR A = 1 TO 10
Replace the first 10 Data Items on Line 470 with;
AA1,A,A9,+,.,A0,9A,1A,00,B
Run the String Sort again then compare the resultant Sorted Data List to the ASCII Code numbers of each Item.

SORTED LIST.

+,.,00,1A,9A,A,A0,A9,AA,B

NOTE! That 'A' appears before 'A0' in the Sorted list.

USES.

The String Sort is used to Sort Lists of Names, Town Names, Magazine Articles and similar alphabetical Information. It will not Sort numbers into their correct numeric order unless some constraints are placed on the format of the Data which is to be placed in the Data List. The numbers must all have the same number of characters. If a decimal point is to be included in the number then it must be in the same position in all Data Items. Two number systems which can be sorted by a String Sort are Post Codes and Telephone Numbers. Another use of a String Sort is for Sorting Receipt Numbers which often include combinations of alpha and numeric characters.
Try replacing the first 10 Data Items on line 470 with 10 Names and Run the programme. Then try some Post Code like numbers and then some Receipt type mixtures of alphas and numerals.

INDEXING.

The use of Multi-dimensional and Indexing Arrays with Sort Routines Dramatically broadens the programming possiblities of Sort Routines. Before looking at Sort Routines which use Indexing Arrays a few words on how Indexing Arrays work is in order. The first example is a single dimensional Array with an Index Array. Assume that the following Data has been entered into your Computer into two single dimension Arays TEST(4) and INDEX(4).

WHEN	TEST(A)	DATA	INDEX(A)	DATA
A=1	TEST(1)	50	INDEX(1)	1
A=2	TEST(2)	87	INDEX(2)	2
A=3	TEST(3)	22	INDEX(3)	3
A=4	TEST(4)	65	INDEX(4)	4

This Data could be Printed to the Screen by the following program.

```
 1 FOR A = 1 TO 4
 2 PRINT TAB(A*A);TEST(INDEX(A));
 3 NEXT A
```

The Screen Display would be;
50 87 22 65
NOTE that the Data is Displayed in the same order as it is stored in Array TEST(4). The FOR/NEXT loop performs 4 passes, the following is a pass by pass description of the operation on LIne 2.

```
FIRST PASS     A=1
SUBSTITUTE FOR A.
       PRINT TAB(1*1);TEST(INDEX(1))
SUBSTITUTE FOR INDEX(1)=1
       PRINT TAB(1);TEST(1)
SUBSTITUTE FOR TEST(1)
       PRINT 50

SECOND PASS    A=2
SUBSTITUTE FOR A.
       PRINT TAB(2*2);TEST(INDEX(2))
SUBSTITUTE FOR INDEX(2)=2
       PRINT TAB(4);TEST(2)
SUBSTITUTE FOR TEST(2)
       PRINT 87

THIRD PASS     A=3
       PRINT TAB(3*3);TEST(INDEX(3))
       PRINT TAB(9);TEST(3)
       PRINT 22

FOURTH PASS    A=4
       PRINT TAB(4*4);TEST(INDEX(4))
       PRINT TAB(16);TEST(4)
       PRINT 65
```

The Index Array has been used inside the TEST Array to determine which Data Item was Printed. If the Data in INDEX(A) was re-arranged by a Sort Routine as follows;

```
       INDEX(A)   DATA
A=1    INDEX(1)   2
A=2    INDEX(2)   4
A=3    INDEX(3)   1
A=4    INDEX(4)   3
```

The above programme would produce the Screen display;

87 65 50 22

The FOR/NEXT loop again performs 4 passes.The following is a description of the first two passes on Line 2.

```
FIRST PASS    A=1
SUBSTITUTE FOR A
      PRINT TAB(1*1);TEST(INDEX(1))
SUBSTITUTE FOR INDEX(1)=2
      PRINT TAB(1);TEST(2)
SUBSTITUTE FOR TEST(2)
      PRINT 87

SECOND PASS A=2
SUBSTITUTE FOR A
      PRINT TAB(2*2);TEST(INDEX(2))
SUBSTITUTE FOR INDEX(2)=4
      PRINT TAB(4);TEST(4)
SUBSTITUTE FOR TEST(4)
      PRINT 65
```

And so on for the third and fourth passes with INDEX(A) determining which value from TEST(A) is Printed. The Array INDEX(A) can be used exactly like an Index for the Array TEST(A) to "LOOK UP" a value in TEST(A). If the third value in TEST(A) was needed the value stored in INDEX(3) is "LOOKED UP". This value is then used in TEST(A) to obtain the third value in order in the Data List stored in TEST(A).

MULTI-DIMENSIONAL ARRAYS

The INDEX Array as shown above for a single dimensional Array can also be used with a Multi-Dimensional Array. Assume that the following Data has been stored in two Arrays in the Computer TEST(A,B) and INDEX(A).

```
      B=1 DATA    B=2 DATA    B=3 DATA
A=1 TEST(1,1)=50 TEST(1,2)=75 TEST(1,3)=3  INDEX(1)=1
A=2 TEST(2,1)=87 TEST(2,2)=10 TEST(2,3)=96 INDEX(2)=2
A=3 TEST(3,1)=22 TEST(3,2)=58 TEST(3,3)=2  INDEX(3)=3
```

This Data could be Printed to Screen by the following programme.

```
 1 FOR A=1 TO 3
 2 FOR B=1 TO 3
 3 PRINT TAB(B*B);TEST(INDEX(A),B));
 4 NEXT B
 5 NEXT A
```

The Screen Display would be

```
    50      75     3
    87      10    96
    22      58     2
```

The Print line can be analysed using the Substitution method.
For the first pass when A=1 B=1

```
SUBSTITUTE FOR A=1
    PRINT TAB(B*B);TEST(INDEX(1),B)
SUBSTITUTE FOR B=1
    PRINT TAB(1*1);TEST(INDEX(1),1)
SUBSTITUTE FOR INDEX(1)=1
    PRINT TAB(1);TEST(1,1)
SUBSTITUTE FOR TEST(1,1)=50
    PRINT  50
```

The whole of the Printing process can be followed using this method. It is Laborious but for our learners it is a sure fire way of geting to know what is happening.

If the Data stored in INDEX(A) was re-arranged by a Sort Routine as below;

```
A=1 INDEX(1)=2
A=2 INDEX(2)=1
A=3 INDEX(3)=3
```

Running the above programme would produce the screen display;

```
87  10  96
50  75   3
22  58  74
```

Note that the first column is now in descending numerical order and that the original relationship across the Data rows has been retained! The substitution method can be used again to help understand what is happening. What would be the correct order of Data in INDEX(A) to have the second column in correct numerical oder?. The correct order is at the end of these notes.

SORTING!

Now to a Sort Routine. Having almost consumed all of the space available this month Shell Sort and others will have to wait until next month. The Sort included this month is very interesting, and compares favourably with Bubble Sort.

SORT BY COUNTING.

Load the original Bubble sort into your computer and retype the following lines;

```
220 DIM S(100),INDEX(100)
260 PRINT S(A);
300 FOR OUTER=1 TO 100
310 COUNT=1
320 FOR INNER=1 TO 100
330 IF S(OUTER)<=S(INNER) THEN 350
340 COUNT=COUNT+1
350 IF S(OUTER)<>S(INNER) THEN 380
360 IF OUTER>=INNER THEN 380
370 COUNT=COUNT+1
380 NEXT INNER
390 INDEX(COUNT)=OUTER
400 NEXT OUTER
410 PRINT S(INDEX(A));
```

Firstly you will note that there is no swap routine as in Bubble Sort. The Data in Array S(100) is not re-aranged by the Sort Routine. Instead the Array INDEX(100) is used to store the correct numerical order of the Data. The Data item number OUTER is stored in INDEX(100) at the position determined by COUNT on Line 390.

Each Data Item is compared with all other Data Items. The Loop OUTER indicates the Data Item number for comparison and Loop INNER steps the Comparison (LINE 330) down the Data list. Each time a Data Item less than the test Data is found, COUNT is incremented by 1. Line 350,360 and 370 increments COUNT if a similar Data Item is found in the Data list ahead of the position of the test Data Item.

Run the Sort and record the time taken to Sort the 100 Data Items. The result when I ran it was;

4 min 40 seconds.

That is interesting, this Sort has two FOR/NEXT LOOPs as has Bubble but why only 4min 40 secs?. This Sort doesnot have the Variable swap routine which Bubble does, and there-in lies the secret! This routine simply COUNTS thus saving quite a bit of time when running. A bonus is that the original Data sequence in S(100) is left intact. The routine doesnot have a Sorted list flag, therefore it would have to proceed through both FOR/NEXT LOOPS even with a previously Sorted Data list. Modify the FOR/NEXT LOOPS to decrease the size of the Data list to 40.(Change Limit from 100 to 40). Run the programme again and note the Run time. For the Data List of 40, Line 330 makes only 1600 comparisons while for a Data List of 100 Line 330 makes 10,000 comparisons. As you can see this Sort Routine is very useful for short Data Lists. Because the routine makes N*N comparisons the run time increases as the square of the Data List length.

CONCLUSION

All the Sort Routines in this set of Articles can be changed to String Sorts by adding '$' to the variables names. Because of the comparison proceedure as described for String Sorts, the RUN time for a String Sort is generally longer than for a Numeric Sort of similar Data length. The length of the Strings being handled also effects Run time. The Sorts presented can be converted to INDEXed Sorts. The final article will be a Listing of all the Sorts with their various possible configurations, all of which will be availble through the Club Library.

The correct Data sequence for the INDEX is 2,3,1. Next Month, a few Shells, Quickies, oddities and if I have time a Forth and machine code Sorts.

LEARNING FORTH

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 22450 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!

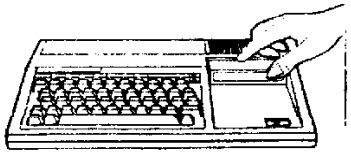
3.EXPERIMENT.

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 sponatanously 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 it 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 ." FIO" 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:
2 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 0 90 20 SMOVE which will copy the first 20 screens (0-19) containing the error messages and the boot screens and the binary core of forth onto screens 0-19 of your new disk

4. IF YOU HAVE ONE DRIVE:
Place the MASTER DISK COPY in Drive 1 and type:
0 DISK_LO ! then press ENTER : GETBLOCK DO I BLOCK UPDATE LOOP ; then press ENTER
5 0 GETBLOCK then ENTER wait till loaded then insert copy disk and type FLUSH Reinsert Master disk after each FLUSH.
10 5 GETBLOCK then ENTER Reinsert Copy disk and FLUSH. 15 10 GETBLOCK then ENTER Reinsert copy disk and FLUSH 20 15 GETBLOCK then ENTER Reinsert copy disk and FLUSH This places the Screens 0-19 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 20 BSAVE then press ENTER. This saves all your above options on disk from SCR#20 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 flash your changes to DISK
8. If you wish to change you disk name for identification with the manager just alter the first 10 characters on SCR# 0 Eg to
FIG-FORTH and FLUSH. 9 Type cold to reboot and see the difference in speed
10. Finally make a backupcopy, tape over the write notch and store in a safe place!

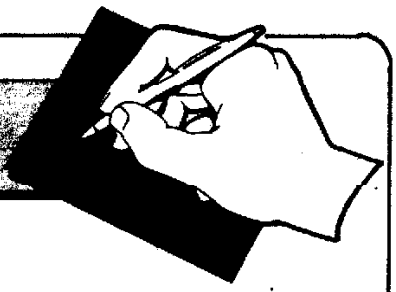
Since this takes up so little space you can easily make a copy of this to use when developing programs on the same disk, so that when you make mistakes like me and have to re-boot, you won't blow your transistors waiting interminably every time you have to reboot and recompile your options.

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.

LOGO

LOGO is probably the least used of all the languages available on the TI99/4A.
According to our recently conducted survey several members have the LOGO 2 software. If we can generate sufficient interest in the language perhaps we may be able to organise regular classes. Or perhaps a Workshop.
In the meantime anyone interested in learning LOGO would be advised to buy a copy of the July Issue of Australian Personal Computer which has just started to run a teach yourself LOGO series titled "PROCEED WITH LOGO" by Harvey Mellar.
The series will be in six parts and if the first episode is any indication, should be quite informative.

Letters to the Editor



THE EDITOR HV99'ers

Please find enclosed a programme which may be suitable for our club magazine.

MULTI-WAY MATHS.

The following programme may be of interest to members who have school children or to school children who want to brush up on their maths.
The programme is written in a fixed manner as it was a Technical College Assignment. Feel free to modify, condense or whatever you may desire.

The programme can be altered to suit any range of maths ability simply by increasing or decreasing the range of random numbers in each of the four sub programmes.
I have found my 7 year old can cope with INT(RND*10)+1 and a friends 13 year old with INT(RND*60)+1, etc.
So here's hoping other children will enjoy maths for a change! GOOD LUCK

GEOFF DANIELS
RATHMINES

```
100 REM MULTI-WAY MATHS
110 REM ####################
120 REM #      BY          #
130 REM #  GEOFF  DANIELS   #
140 REM #    H.V. 99'ers.  #
150 REM ####################
160 CALL CLEAR
170 PRINT "INTRODUCTION"
180 PRINT
190 PRINT "#1 ADDITION"
200 PRINT
210 PRINT "#2 SUBTRACTION"
220 PRINT
230 PRINT "#3 MULTIPLICATION
"
240 PRINT
250 PRINT "#4 DIVISION"
260 PRINT
270 PRINT "#5 QUIT"
280 INPUT N
290 IF N>5 THEN 300 ELSE 340
300 PRINT "STUPID TWIT,ONLY
1 TO 5"
310 FOR DELAY=1 TO 250
320 NEXT DELAY
330 GOTO 160
340 ON N GOTO 1000,5000,1000
0,15000,20000
1000 PRINT "ADDITION"
1010 PRINT
1020 PRINT "I'M THINKING"
1030 FOR DELAY=1 TO 100
1040 NEXT DELAY
1050 RANDOMIZE
1060 A=INT(RND*20)+1
1070 B=INT(RND*20)+1
1080 C=B+A
1090 PRINT
1100 PRINT A
1110 PRINT "+"
1120 PRINT B
1130 PRINT "------"
1140 INPUT D
1150 IF D<>C THEN 1160 ELSE
1180
1160 PRINT "FOOL, YOU GOT IT
 WRONG"
1170 GOTO 1070
1180 PRINT "TERRIFIC, YOU AR
E CORRECT"
1190 FOR DELAY=1 TO 750
1200 NEXT DELAY
1210 INPUT "SAME AGAIN?(Y/N)
":T$
1220 IF T$<>"N" THEN 1010 EL
SE 160
5000 PRINT
5010 PRINT "I'M THINKING"
5020 FOR DELAY=1 TO 100
5030 NEXT DELAY
5040 REM
5050 RANDOMIZE
5060 A=INT(RND*20)+1
5070 B=INT(RND*20)+1
5080 C=A-B
5090 IF C<1 THEN 5060 ELSE 5
110
5100 PRINT
5110 PRINT A
5120 PRINT "-"
5130 PRINT B
5140 PRINT "------"
5150 INPUT D
5160 IF D<>C THEN 5170 ELSE
5190
5170 PRINT "IDIOT, YOU ARE W
RONG"
5180 GOTO 5100
5190 PRINT "GREAT, YOU ARE R
IGHT"
5200 FOR DELAY=1 TO 750
5210 NEXT DELAY
5220 INPUT "SAME AGAIN? (Y/N
)":T$
10000 PRINT "MULTIPLICATION"
10010 PRINT
10020 PRINT "I'M THINKING"
10030 FOR DELAY=1 TO 100
10040 NEXT DELAY
10050 RANDOMIZE
10060 A=INT(RND*20)+1
10070 B=INT(RND*20)+1
10080 C=A*B
10090 PRINT
10100 PRINT A
10110 PRINT "*"
10120 PRINT B
10130 PRINT "------"
10140 INPUT D
10150 IF D<>C THEN 10160 ELS
E 10180
10160 PRINT "DIPSTICK,THAT I
SN'T RIGHT"
10170 GOTO 10090
10180 PRINT "BONZA COBBER YO
U DID IT"
10190 FOR DELAY=1 TO 750
10200 NEXT DELAY
10210 INPUT "SAME AGAIN? (Y/
N)":T$
10220 IF T$<>"N" THEN 10010
ELSE 160
15000 PRINT "DIVISION"
15010 PRINT
15020 PRINT "I'M THINKING"
15030 FOR DELAY=1 TO 100
15040 NEXT DELAY
15050 RANDOMIZE
15060 A=INT(RND*20)+1
15070 B=INT(RND*20)+1
15080 C=A/B
15090 F=INT(C)
15100 IF F<>C THEN 15060 ELS
E 15110
15110 IF C<1 THEN 15060
15120 PRINT
15130 PRINT A
15140 PRINT "/"
15150 PRINT B
15160 PRINT "------"
15170 INPUT D
15180 IF D<>F THEN 15190 ELS
E 15210
15190 PRINT "AGAIN WITH OUT
THE MISTAKES"
15200 GOTO 15120
15210 PRINT "CORRECT, YOU AR
E A GENIUS"
15220 FOR DELAY=1 TO 750
15230 NEXT DELAY
15240 INPUT "SAME AGAIN (Y/N
)":T$
15250 IF T$<>"N" THEN 15010
ELSE 160
20000 CALL CLEAR
20010 PRINT "THATS IT"
20020 FOR DELAY=1 TO 250
20030 NEXT DELAY
20040 PRINT
20050 PRINT " YOU HAVE ESCAP
ED FROM"
20060 PRINT "THE DREADED MAT
HS MONSTER"
20070 FOR DELAY=1 TO 500
20080 NEXT DELAY
20090 PRINT
20100 PRINT "GOOD"
20110 PRINT "   B"
20120 PRINT "    Y"
20130 PRINT "     E"
20140 PRINT "     E"
20150 PRINT "     E"
20160 PRINT "      E"
20170 PRINT "       E"
20180 PRINT "        E"
20190 PRINT "         EEEEE
"
20200 END
```

EXTENDED TUTORIAL NO.7

FUNNELWEB FARM

It's time once again to get back to the regular Tutorial material, continuing with the ways and means of scrunching program length. As I remarked before, it's a subject I'm not completely comfortable talking about because, while this series has been devoted to better XBasic programming, most things you can do to scrunch Basic programs make them less readable by ordinary mortals, given reasonable programming skill in the first place. The other reason for my reluctance is that this kind of discussion tends to degenerate into a collection of unrelated items, yet another set of "Tips", when I really want this series to be a gentle but systematic look at the workings of the machine and its language(s).

Anyway let's start at the small end of things and work up to the larger scale. Last time we looked at the space taken by simple variables. The most obvious thing is to keep variable names short. I don't recommend this until late in the piece because it is such a cheap and obvious way of gaining bytes that you might as well have the help of descriptive variable names until you are absolutely desperate for bytes. Absolute desperation has not occurred until you have had several rounds of byte saving already. The shortest variable name has only one letter character, but TI Basics also officially allow "" (shift-2) and "_" (fctn-U) as variable names. It has to be a fairly long SUBprogram before you need more than 26 simple numeric variables but it can happen. On this console there are 3 other single characters which can be used as variable names. Experiment to find if they exist on your machine. The nagging problem is that they are not documented.

There is another way to use variable names to shorten a program. Remember from last time that a one digit numeric constant is treated as a string and takes 3 bytes, while a single letter variable takes only 1 byte. If a particular numeric value occurs frequently in a SUBprogram, Ø or 1 being common examples, then it may be worth the overhead, 14 bytes plus the defining statement, for a new variable of that value if you can then save 2 bytes on numerous occasions. A frequently used longer numeric constant, as might occur in CHAR or SPRITE manipulations, yields more bytes each time. It is a matter of doing careful book-keeping and byte counting in each SUBprogram. Once you start down this track be alert for further gains -- if you have defined S=7 and F=5 then it saves a byte to write S*F instead of 35. If you can reuse an already defined variable name then the investment is paid back faster, but this requires keeping very careful track of program flow. Go back to the example of a Key/Joystick routine in an earlier Tutorial and see if you can shorten it by reducing the number of variables used.

Replacement of numbers by variables has precedents in other languages. In TI-Forth the numbers Ø,1,2,3 are not treated directly as numbers but are defined words in the language.

There is another little way that cunning entry of characters can shorten programs. This is in the entry of graphics characters with ASCII values above 127 in the upper color groups of XB by writing strings with DISPLAY AT instead of H VCHAR CALLs. Characters in this range can be entered in strings in program statements by use of the CRTL key, rather than by using the CHR$ function. It does tend to make the program incomprehensible as these echo as blanks to the screen. They will appear with their defined shapes if the line is called up for editing after RUNning the program. These codes are also used as XB tokens and can only be used within strings. I should add in passing that I am in total agreement with the TI designers' choice not to allow abbreviated (direct token) entry of Basic keywords. If you want that sort of thing you should be back on your Sinclair or Commodore, and you

probably don't believe in relocatable object files either,

The use of arrays to represent small collections of numbers needs detailed working out. The gains from less variable table overhead and simplified parameter passing to SUBprograms have to be balanced against the extra bytes needed for each program reference. Let the program logic be your initial guide.

This idea of using fewer bytes to represent quantities leads on to the larger subject of data compaction. One byte can carry 256 different values, and one third to one half of those can be conveniently entered from the keyboard. It's sheer overkill to use an 8 byte floating point number to represent just a few values, or even just a logical (Boolean) variable which really needs only one bit. Some languages compact Boolean variables as bits in a word or words. The CRU single bit bus of the TMS-9900 provides an ideal mechanism for bit storage and testing, but as in so many other areas the 99/4a hardware does not do justice to its CPU. The later TMS-9995 in fact has a little on-board CRU memory for just this purpose.
Opportunities for data compaction are limited in XB both because of the structure of the language (it has only character strings, floating point numerics and arrays of these as data types) and the convoluted, slow way it is implemented via GROMs and VDP memory. Any scheme for coding or compacting needs computation to pack and unpack the data. At the machine code level the tradeoffs between memory use and speed are different from those in Basic, especially TI-99 Basics, because Basic is so much slower. In my experience the use of string variables to compact data in active parts of a program is almost always doomed to failure because of slow string handling by XB and pauses for garbage collection. Data compaction can be useful though in setting up initial graphics designs or for music data. There are only so many different notes, in pitch length and volume used in any given short musical piece, and since each note takes time to play and is handled by the machine on an interrupt driven basis, this time can be used to do the computations needed to unravel the data for the next note.

Let's have a look at the graphics screen example. Suppose that in setting up a game screen, either one of two characters, maybe the same pattern in two different color groups, has to be written to 20 locations in various parts of the screen. The simplest way is a whole succession of CALL HCHARs - assuming the display is not suited to generation with DISPLAY ATs - and that's the way you will find it done in many programs (just like long lists of CALL SOUNDs). What is totally unforgivable is to find incompetent magazine or commercial programs with inefficient coding that force inconveniences like CALL FILES(1) on the user.

```
1000  CALL HCHAR(23,12,105) 1010 CALL
HCHAR etc etc
```

This takes over 600 bytes. How can it be shortened ? One way, a bit of a dead end in this example, is to use multi-statement lines. This would be shorter by 30 bytes or so, and marginally faster. The real improvement is to eliminate the repetition of CALL HCHAR - remember CALL is cheap but HCHAR is expensive - by using a loop and DATA statements.

```
1000 FOR I=1 TO 20 :: READ A
,B,C :: CALL HCHAR(A,B,C)::
NEXT I
1010 DATA 23,12,105, etc etc
```

Now all but one of those HCHARs have gone. The price paid is loop and DATA execution overhead and the increased possibilities for clerical errors since the DATA items have been divorced from their proper context. At this stage you may be feeling very pleased with yourself, but then you find that to add another feature to your program you need more space. Now is the time to reflect seriously on data compression. A column index for HCHAR can only have the values 1 to 32 and rows 1 to 24. One of these values can be expressed by 1 byte with possibilities to burn. Say you use 1 byte for each row or column value then. Expressing the bytes efficiently as DATA is the next problem - there are a few bytes of overhead for each item in a DATA list, and DATA lists of a lot of short items are notorious for causing a "line too long" error. So let's pack them in a single string and use SEG$ to unpack them, with ASC to turn

a ASCII character back to a value for CHAR. A minor problem is that characters 1 to 32 can't be entered directly in XB, so just use characters starting with "A" and subtract 64. The opposite problem may occur with the string for the character values if upper graphics sets are being used. Then just use lower values and add a correction. So now the code might look like

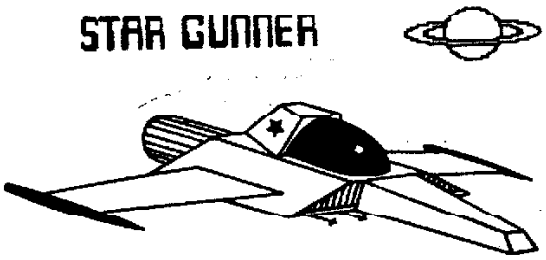
```
100 READ A$,B$,C$ :: FOR I=
1 TO 20 :: CALL HCHAR(ASC(SE
G$(A$,I,1))-64,ASC(SEG$(B$,I
1))-64,ASC(SEG$(C$,I,1)+32):
: NEXT I
110 DATA "W...  ","L...  ","I.
```

You could further pack the data into a single string and modify the SEG$ statements accordingly, but it might not be worth it. Remember now that the problem posed involved writing only two different characters and work out how you could compact things still further for this limited case. This example is based on one of methods that was used to squeeze TXB into console memory. An extreme example of data compression comes when the data is regular enough that it can be generated by a formula or procedure. This is something that has to be worked out in each case.

The use of loops as in the examples above applies in other situations, particularly in CHAR definitions. XB allows the use of multiple arguments in CHAR, COLOR, SPRITE and suchlike SUBprograms. This is better and faster than using individual SUBprogram CALLs for each item in the list. The real dilemma comes when we try to use a loop to compact the program further. Critical parts of the program may be slowed down unacceptably so that you may find yourself using compact slow code in some parts of a program and longer but faster forms elsewhere. Just in passing I should remind you to null out on exit from a SUBprogram, any string variables not required to keep their value till the next CALL. This particularly applies to string variables used for READ, INPUT, PRINT etc operations involving long strings. Remember that it is the length of a program while RUNning that really counts.

Time to sign off for this issue now. Next Tutorial will continue with more aspects of byte saving.

STAR GUNNER



BY JOHN SMART
****

This game places you in the gun turret on the "Odysseus", a merchant ship being raided by a squadron of TI (Techani Imperium) Fighters. They fly around the ship launching missiles and dodging your anti-aircraft fire.

As the gunner you must shoot down the TI fighters by using Joystick #1 to move the sight around the screen and fire the guns. When the guns are fired they flare and the shell detonates near the sights. However, the TI fighters return fire with missiles which gradually grow larger as they approach. If you fail to shoot them down they will hit the ship. On the fifth hit the ship will be destroyed.

As the game progresses the missiles are fired more rapidly. Techani fighters(see below) also appear more frequently.

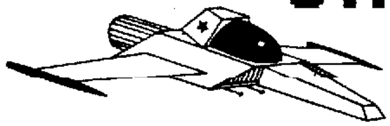
LEVELS OF DIFFICULTY>
There are five levels of difficulty, ranging from "Easy" to "Masochistic". The speed and frequency of the missiles, and the frequency of the Techani Fighters is directly proportional to the level of difficulty.

TECHANI FIGHTERS.
The Techani fighters appear as multi-coloured flashing missiles approaching at EXTREMELY high speeds. A warning message appears directly before one approaches.
SCORING.

TI Fighters.....	10 Points
Missile.....	5 Points
Techani fighters.....	50 Points

N.B. The Techani fighters will do twice the damage of a single missile and this has prompted some cynics to claim that the only purpose of the craft is to reduce the length of the game, this of course is not true (would we lie to you?)

STAR GUNNER

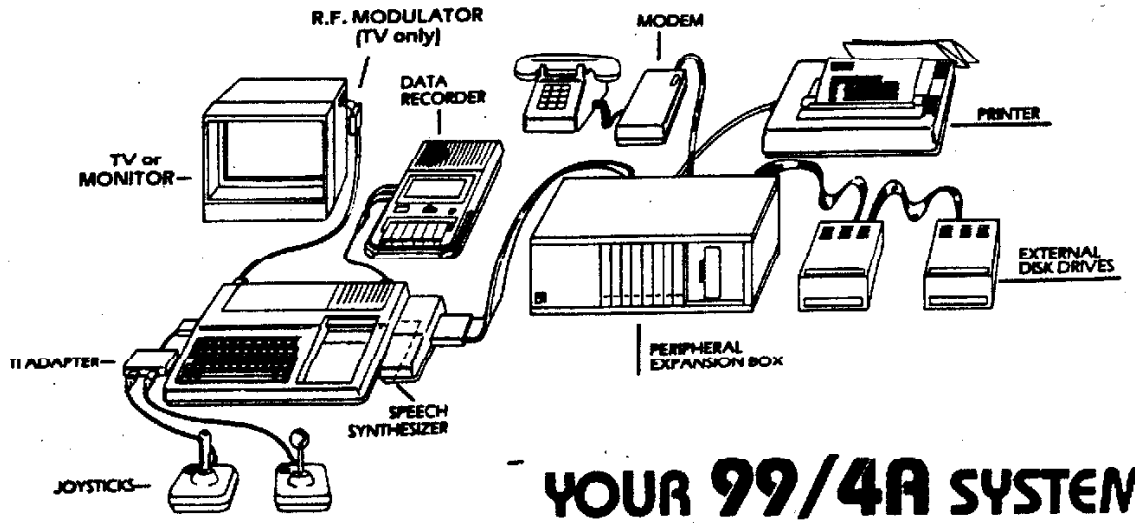



```
10 !This prog.donated to
    HUNTER VALLEY 99'ERS
    P.D library to publish
    in the newsletter.
100 !**********************
*
110 !*
*
120 !* S T A R   G U N N E R
*
130 !*      By John Smart
*    *      & Iain Holmes
*
140 !*
*    **********************
*
150 !
160 ! Use Joysticks to move
      the sight around the
      screen to shoot down
      the T.I fighters and
      their missiles.Five
170 ! Missile hits will
      your ship.Also beware
      of the dreaded Techani
      fighters,which look
      like multi-coloured
180 ! missiles,but which do
      twice the damage of
      ordinary missile hits.
190 !
200 ! P.S. If you don't want
      to type this listing,
      you can snet $2.00 wit
h     a blank tape and a
      stamped,self addressed
210 ! envelope to the follow
-     ing address:
       'John Smart
        22 Northcott Ave
        East Maitland
        N.S.W.  2323
220 !
230 ! P.P.S.Other even bette
r     games are available
      from the above addesss
,     including
240 !STARLORD:Exiting space
      action where you fly
      a mercenary starship.
    STAR GUNNER II:Pilot
      your Star Fighter
250 ! down a 3D trench
      through a fierce
      barrage of missiles
      to torpedoe the silo.
260 !MERLIN FILE:A versitile
      tape based filing
      program for a unexpa-
      nded TI.
270 !ASTEROID FIELD:Shoot
      down asteroids in an
      asteroid field.
    !CON-SHIP:Design your
      own starship and then
280 ! relieve the monotony
      by blasting your
      opponent into nul
      valued bits.
290 !GRAND PRIX:A racing
      game.
    !STAR PIRATE:A 3D arcade
      adventure in which
      you are a star pirate
300 ! making a living by
      crippling and robbing
      star merchants.But
      watch out for the
      Galactic police,who
310 ! are after the bounty
      on your head!

    For a complete catalog
    write to the above
320 ! address or phone
      (049) 336003.
    P.P.P.S. You don't have
    to type in these REM
    statements if you don't
    want to.
330 CALL CHAR(108,"01010000
0000000000000000000000000000
0000000000000000000000000000
00000000000000000000")
340 ON WARNING NEXT
350 PT=116
360 SP=2
370 RANDOMIZE :: CALL SCREEN
(2):: CALL MAGNIFY(3)
380 CALL TITLE(SP,ATT,SPI)
390 CALL CHAR(112,"10204080
183C6FDC6838180402010001000
40202820C67EC68202020408010")
400 CALL CHAR(116,"00000000
0000001030100000000000000000
0000000000000800000000000")
410 CALL CHAR(120,"00000000
00001030703010000000000000000
000000000080C080")
420 CALL CHAR(124,"00000000
10103071E070301010000000000000
0000000080C0F0C080")
430 CALL CHAR(128,"00000010
1030700E3C0E070301010000000000
0000000C0E078E0C080")
440 CALL CHAR(132,"00010101
3070E1CF81C0E0703010101000000
0000C0E0703E70E0C000")
450 CALL CHAR(136,"00402414
C3C000000003C0C1424400801722
4283003C000000003C3028240201"
)
460 CALL CHAR(140,"01000220
8002A00441104200A00220000200
00201A004A004100840124001000"
)
470 CALL CHAR(100,"C0A050281
40B0707",101,"03050A1428D0E0
E0",104,"00402018100F070F",1
05,"0002041BD8F0E0F0")
480 CALL CLEAR :: CALL SCREE
N(2):: FOR C=0 TO 7 :: CALL
COLOR(C,3,1):: NEXT C

490 ! NOTHING HERE
500 FOR ST=1 TO 12 :: CALL H
CHAR(INT(RND*24)+1,INT(RND*3
2)+1,46):: NEXT ST
510 CALL HCHAR(24,2,101):: C
ALL HCHAR(24,31,100):: CALL
HCHAR(23,3,105):: CALL HCHAR
(23,30,104):: CALL COLOR(9,1
5,1)
520 SC,HITS,ACC=0
530 DISPLAY AT(1,1):"SCORE:"
;SC :: DISPLAY AT(2,1):"HITS
:";HITS :: DISPLAY AT(3,1):"
HIGH SCORE:";HSC
540 CALL SPRITE(#5,112,INT(R
ND*13)+3,INT(RND*191)+1,INT(
RND*255)+1)
550 CALL SPRITE(#1,136,11,96
,128)
560 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
570 DISPLAY AT(1,1):"SCORE:"
;SC :: DISPLAY AT(2,1):"HITS
:";HITS :: DISPLAY AT(3,1):"
HIGH SCORE:";HSC
580 X1,Y1=0 :: ACC1=INT(RND*
8)-4 :: ACC2=INT(RND*8)-4
590 T1=INT(RND*10)+1 :: FOR
T=1 TO T1
600 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
610 IF RND*ATT(1 AND MS=0 TH
EN GOSUB 860 ELSE IF MS=1 TH
EN PT=PT+SP :: CALL PATTERN(
#4,PT):: IF PT)132 THEN 890
620 CALL MOTION(#5,X1,Y1)::
X1=X1+ACC1 :: Y1=Y1+ACC2 ::
IF ABS(X1))127 OR ABS(Y1))12
7 THEN GOSUB 970
630 NEXT T
640 FOR T=1 TO T1
650 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
660 IF RND*ATT(1 AND MS=0 TH
EN GOSUB 860 ELSE IF MS=1 TH
EN PT=PT+2 :: CALL PATTERN(#
4,PT):: IF PT)132 THEN 890
670 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
680 CALL MOTION(#5,X1,Y1)::
X1=X1-ACC1 :: Y1=Y1-ACC2 ::
IF ABS(X1))127 OR ABS(Y1))12
7 THEN GOSUB 970
690 CALL MOTION(#5,X1,Y1)::
X1=X1-ACC1 :: Y1=Y1-ACC2 ::
IF ABS(X1))127 OR ABS(Y1))12
7 THEN GOSUB 970
700 NEXT T
710 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
720 IF RND*ATT(1 AND MS=0 TH
EN GOSUB 860 ELSE IF MS=1 TH
EN PT=PT+2 :: CALL PATTERN(#
4,PT):: IF PT)132 THEN 890
730 CALL MOTION(#5,X1,Y1)::
X1=X1+ACC1 :: Y1=Y1+ACC2 ::
IF ABS(X1))127 OR ABS(Y1))12
7 THEN GOSUB 970
740 FOR T=1 TO T1
750 CALL JOYST(1,X,Y):: CALL
 MOTION(#1,-Y*8,X*8):: CALL
KEY(1,K,S):: IF K=18 THEN GO
SUB 880
760 IF RND*ATT(1 AND MS=0 TH
EN GOSUB 860 ELSE IF MS=1 TH
EN PT=PT+2 :: CALL PATTERN(#
4,PT):: IF PT)132 THEN 890
770 CALL MOTION(#5,X1,Y1)::
X1=X1+ACC1 :: Y1=Y1+ACC2 ::
IF ABS(X1))127 OR ABS(Y1))12
7 THEN GOSUB 970
780 NEXT T
790 GOTO 580
```

```
800 CALL SOUND(-100,-6,0):: CALL COLOR(10,9,1):: CALL COLOR(10,2,1)
810 CALL POSITION(#1,X2,Y2):: CALL SPRITE(#2,148,9,X2,Y2):: CALL COINC(#2,#4,12,T)
820 IF T=-1 THEN 850 ELSE CALL COINC(#1,#5,12,T):: CALL DELSPRITE(#2):: IF T=0 THEN RETURN
830 CALL DELSPRITE(#2):: ATT=ATT-1 :: CALL BANG(5):: SP=SP+SPI :: SC=SC+10 :: IF RND#ATT<1 THEN GOSUB 1000
840 GOTO 540
850 SC=SC+5 :: DISPLAY AT(1,1):"SCORE:";SC :: PT=116 :: M$=0 :: CALL BANG(4):: CALL DELSPRITE(#2):: RETURN
860 CALL MOTION(#1,0,0):: DISPLAY AT(12,12):"MISSILE!" :: FOR D=1 TO 3 :: CALL SOUND(-50,750,0):: FOR D=1 TO 50 :: NEXT D :: NEXT D
870 DISPLAY AT(12,12):
880 M$=1 :: CALL POSITION(#3,X3,Y3):: CALL SPRITE(#4,116,16,X3,Y3):: RETURN
890 M$=0 :: PT=116 :: CALL SCREEN(16):: CALL SOUND(-2500,-7,0):: CALL DELSPRITE(#4,#6):: HITS=HITS+1 :: CALL SCREEN(2)
900 IF HITS<5 AND TCF=0 THEN 550 ELSE IF HITS<5 AND TCF=1 THEN TCF=0 :: GOTO 540
910 DISPLAY AT(2,1):"HITS:";HITS :: CALL DELSPRITE(ALL):: DISPLAY AT(12,10):"GAME OVER"
920 IF SC>HSC THEN 930 ELSE 940
930 HSC=SC :: DISPLAY AT(3,12):HSC :: FOR D=1 TO 3 :: CALL SOUND(-50,1000,0):: DISPLAY AT(3,12):"" :: FOR D=1 TO 50 :: NEXT D :: DISPLAY AT(3,12):HSC :: NEXT D
940 M$='HIT ENTER TO PLAY AGAIN.....'
950 M$=SEG$(M$,LEN(M$)-1,1)&SEG$(M$,1,LEN(M$)-1):: DISPLAY AT(23,1):M$ :: CALL KEY(0,K,S):: IF S=0 THEN 950
960 GOTO 340
970 X1,Y1=0 :: RETURN
980 T=0 :: T1=0 :: X1=0 :: Y1=0
990 GOTO 580
1000 CALL MOTION(#1,0,0):: DISPLAY AT(12,8):"TECHNI FIGHTER" :: FOR D=1 TO 3 :: CALL SOUND(-50,750,0):: FOR D=1 TO 50 :: NEXT D :: NEXT D
1010 DISPLAY AT(12,8):""
1020 CALL SPRITE(#6,116,INT(RND#13)+3,INT(RND#180)+1,INT(RND#256)+1)
1030 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y#8,X#8):: CALL KEY(1,K,S):: IF K=18 THEN GOSUB 1140
1040 FOR T=1 TO ATT :: NEXT T :: CALL PATTERN(#6,128):: CALL COLOR(#6,INT(RND#13)+3)
1050 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y#8,X#8):: CALL KEY(1,K,S):: IF K=18 THEN GOSUB 1140
1060 FOR T=1 TO ATT :: NEXT T :: CALL PATTERN(#6,124):: CALL COLOR(#6,INT(RND#13)+3)
1070 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y#8,X#8):: CALL KEY(1,K,S):: IF K=18 THEN GOSUB 1140
1080 FOR T=1 TO ATT :: NEXT T :: CALL PATTERN(#6,128):: CALL COLOR(#6,INT(RND#13)+3)
1090 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y#8,X#8):: CALL KEY(1,K,S):: IF K=18 THEN GOSUB 1140
1100 FOR T=1 TO ATT :: NEXT T :: CALL PATTERN(#6,132):: CALL COLOR(#6,INT(RND#13)+3)
1110 CALL JOYST(1,X,Y):: CALL MOTION(#1,-Y#8,X#8):: CALL KEY(1,K,S):: IF K=18 THEN GOSUB 1140
1120 CALL POSITION(#6,AA,BB):: IF AA+BB=0 THEN RETURN
1130 CALL DELSPRITE(#6):: CALL SCREEN(2):: CALL SOUND(-100,-6,0):: CALL COLOR(10,9,1):: CALL COLOR(10,2,1)
1150 CALL POSITION(#1,X2,Y2):: CALL SPRITE(#2,148,9,X2,Y2):: CALL COINC(#2,#6,12,T)
1160 IF T=0 THEN CALL DELSPRITE(#2):: RETURN
1170 CALL BANG(6):: SC=SC+30 :: RETURN
1180 SUB TITLE(SP,ATT,SPI)
1190 CALL CLEAR :: CALL SCREEN(3):: CALL CHARSET
1200 FOR C=0 TO 8 :: CALL COLOR(C,16,1)
1210 NEXT C :: DISPLAY AT(10,9):" STAR GUNNER" :: DISPLAY AT(12,9):"BY JOHN SMART" :: DISPLAY AT(14,2):"COPYRIGHT (C) J. SMART 1984"
1220 M$="HIT ENTER TO BEGIN. ....."
1230 M$=SEG$(M$,LEN(M$)-1,1)&SEG$(M$,1,LEN(M$)-1):: DISPLAY AT(23,1):M$ :: CALL KEY(0,K,S):: IF S=0 THEN 1230
1240 CALL CLEAR :: DISPLAY AT(12,1):" PLEASE CHOOSE LEVEL :          1)VERY EASY          2)EASY          3)MEDIUM "
1250 DISPLAY AT(16,1):"  4)HARD                5)DESTATING'
1260 ACCEPT AT(18,5)SIZE(1)BEEP VALIDATE(DIGIT):L
1270 IF L<1 OR L>5 THEN 1260
1280 ON L GOTO 1290,1300,1310,1320,1330
1290 SPI=.05 :: ATT=32 :: SP=.25 :: SUBEXIT
1300 SP=.1 :: ATT=20 :: SP=.5 :: SUBEXIT
1310 SPI=.5 :: ATT=10 :: SP=2 :: SUBEXIT
1320 SPI=2 :: ATT=5 :: SP=4 :: SUBEXIT
1330 SPI=4 :: ATT=1 :: SP=4 :: SUBEXIT
1340 SUBEND
1350 SUB BANG(H)
1360 CALL MOTION(#1,0,0)
1370 CALL POSITION(#H,A1,B1) :: IF A1#B1=0 THEN CALL POSITION(#1,A1,B1)
1380 CALL DELSPRITE(#2,#H):: FOR A=10 TO 17 :: CALL SPRITE(#A,100,16,A1,B1):: NEXT A
1390 CALL SOUND(-1000,-7,0)
1400 CALL MOTION(#10,0,32,#11,1,32,32,#12,32,0,#13,32,-32,#14,0,-32,#15,-32,-32,#16,-32,0,#17,-32,32)
1410 FOR C=16 TO 2 STEP -4 :: CALL COLOR(#10,C,#11,C,#12,C,#13,C,#14,C,#15,C,#16,C,#17,C):: NEXT C
1420 CALL DELSPRITE(#10,#11,#12,#13,#14,#15,#16,#17)
1430 SUBEND
```



YOUR 99/4A SYSTEM

FUNLWRITER

FUNNELWEB FARM and the HUNTER VALLEY 99 USER GROUP now announce the availability of FUNLWRITER as a Public Domain disk. And what on earth is FUNLWRITER ? It is a program that allows TI-Writer to be loaded and run without the TI-Writer module, using the Extended Basic module instead. If you already have TI-Writer this is an insurance against the module getting zapped, and a convenience as XB is likely to be the general purpose module usually resident in the console. If you don't then it means that you can now run TI-Writer without having to fork out the exorbitant price being demanded by Imagic. TI in fact released the revised *fix 1 version of the TI-Writer disk as Public Domain software to User Groups. So what you would be paying for is the manual and a simple cartridge with only one lousy GROM. We can't supply the manual, but we have made the cartridge totally unnecessary and at the same time made a number of improvements to TI-Writer. In fact the program is so little inferior in use in any way at all, and sufficiently superior in a number of ways that the TI-Writer module is now essentially obsolete.

As it is you don't have to pay out a fortune for such programs as one Public Domain version of USA origin has come to Newcastle via the UK. Somehow or other free goodies like that rarely seemed to make it past the great TI-SHUG software and information bottleneck, but HV99 intends to do a lot better for its members and to make real Public Domain contributions to the TI-99/4a user community in its own turn. Oh well, each group to its own style.

I hear you ask, if you have been given one already why bother to write another ? One answer is that it's like the mountain - the challenge is there waiting, and there is nothing like climbing it yourself. Actually we had started coding it before then, and the Newcastle version was already superior. Since then it has been further improved and enhanced and Version 2.1, specifically identified as such, is the current official release. Some earlier versions have been given out, and V2.1 may already be a response to some requests for improvements. The main feature of FUNLWRITER is that Show Directory works, better and faster in fact in FUNLWRITER than in the original.

The only sacrifice made in adapting TI-Writer to the XB module is that the TI-Writer module's selection screen is not available and must be simulated by auto-RUNning of DSK1.LOAD on selection of XB. This file, about 25 sectors long, is the ONLY file other than the TI-Writer *fix I files that needs be on the working disk, and this auto-load is the only extra disk activity over normal TI-Writer operation. We have even managed to eliminate the unnecessary reloading of the Formatter from disk after each use in V 2.1 of FUNLWRITER. LOAD has been kept as short as possible to minimize loading time, the XB disk directory routine provided in earlier versions having been eliminated for this reason. There is also a document file FUNNELDOC which is appended at the end of this article to give all the gory details.

What this work has shown is that TI's programmers never really fully exploited the capabilities of the machine, despite having all the inside information and fancy development systems, and I would say that is true for just about all TI-99 software I have seen, even including FUNLWRITER for which we programmed every last detail here. That is why I have in FUNNELDOC placed emphasis on free exchange of programming ideas among active creators of software. An atmosphere of secretiveness and petty commercialism at the User Group level now only works to the detriment of the TI-99 user community. One thing for sure is that any program for which someone is demanding real money should be of significantly higher quality than FUNLWRITER which is for free.

Some interesting observations came up during this work. One is that the Editor and Formatter were clearly written by different people as the programming styles are quite distinct. Stephen Shaw reminded me that TI-Writer's Editor is a lineal descendant of the P-Code Editor via the E/A Editor. The E/A editor is an old friend but I have never seen the P-System editor. Perhaps if Imagic

had given me a realistic price on P-Code when I called them, I might still be mucking about with Pascal on heaven knows what instead of doing FUNLWRITER in Assembler. The real matter was, judging from the style of the code, written at another time by someone who had taken the prescriptions of the TI-99/4a Technical Data Manual fairly seriously, such as consistent use of indexed addressing with respect to a pointer to the 16-bit PAD. The reformatter also contains code specifically included to prevent its use with the E/A RUN PROGRAM FILE option or with XB for that matter. Only 1 byte needs be changed to let it run this way.

I think the best way to finish off this article is to include the FUNLWRITER document file with its detailed description of, and instructions for running it. The program has run successfully on a 2.2 1983 title screen machine, and with a Corcomp disk controller. We have not checked if the new Show Directory works with DS/DD disks because the particular PE box card has quit working on DD disks. I remain deeply suspicious of Corcomp's engineering and quality standards. Oh well, read on !

FUNNELWEB FARM TI-WRITER LOADER
for use with Extended Basic.
-- ------------------
DF1.FUNLWRITER V 2.1
-- ----------------------------

1. General Notes

This program has been prepared to allow use of the TI-Writer word processor from the Extended Basic module with no loss of functionality, and some minor improvements. It is as convenient to use as the original, and generates as little disk activity as we can manage. It has been prepared as a PUBLIC DOMAIN contribution from the newly formed Hunter Valley 99 User Group to the rest of the TI-99/4a user community. It is neither to be sold nor distributed with excessive copy fees. We only request that when you pass it on that the Funnelweb Farm and Hunter Valley 99 User Group attribution be preserved and mentioned and that you remember us when circulating other full-system software. Please also always transmit this document file along with the loader.

The files on this diskette are a Extended Basic LOAD program, this document file, and the *fix 1 revision of TI-Writer, circulated to User Groups as a public domain release by TI. The manual for TI-Writer remains, to our present knowledge, a copyright item and so you will have to read someone else's if you don't already have TI-Writer. The program was prepared as our own insurance policy on failure of our TI-Writer module, and as protest at the rapacious pricing by Imagic Australia on TI-Writer, and against excessive pricing of inferior commercial offerings in this genre. You will find yourself leaving the TI-Writer module on the shelf if XB is already in the machine, saving wear and tear on the console.

2. User Notes

Before doing anything else, make a working copy of this disk, keeping the original safe as a master copy for backup and for passing on. The only unusual file is LOAD, but this and the other files may be copied with Disk Manager in the normal way. Place the working copy in drive #1 and select Extended Basic. The LOAD program will auto-RUN (or can be called up from XB), and a menu with the usual 3 choices will be presented. The Editor, Formatter, and Utility options load just as quickly as they do from TI-Writer as the loader, written in assembly language, is already in place. When you exit the Editor or Formatter the program returns you to the title screen instead of to the TI-Writer module's menu screen, and it is only necessary to select XB again. There has been NO decrease in the size of files that may be loaded by or written with the Editor.

(a) Editor

The Editor functions in all respects as it does with the TI-Writer module with the following exceptions (all

are improvements)

(i) The color selections using <ctrl-3> may be altered from within the XB LOAD program to suit your taste. This is done in the CALL COLOR statement in line #250. The assembly loader reads the color choices for XB Groups 10-14 and writes these into the Editor as the 5 color selections. The CALL COLOR statement must not be deleted !

(ii) A printer device-name is pre-loaded into the Editor by the loader and may be changed in the Extended Basic program to suit your own needs if different. This is done in the statement in line #250, CALL LINK("EDITA","devicename"). The LOAD program may be edited and reSAVEd, but RESequencing will destroy its integrity.

(iii) Normally the Editor disables the <fctn => system reset key, except when it is in Show Directory, a GPL routine in the TI-Writer module. Now the <quit> key remains disabled at all times.

(iv) An assembly language routine is used for the Show Directory function and handles standard TI SS-SD diskettes with up to 127 filenames. The paged presentation is generally easier to use than the one-time scrolled original version. Paging is controlled by the use of the single <ctrl> or <shift> keys. Pressing <fctn> causes the program to check and indicate the type of program files on the disk.

(v) An asterix after the file length shown in the directory indicates a fractured file, one stored in more than one contiguous block on the disk. This may of be value outside TI-Writer when preparing frequently used disks, to reduce disk head activity in use.

(b) Formatter

The Formatter functions normally with the following exceptions (again all are improvements on the original).

(i) The printer device-name is specified in the statement CALL LINK("FORMA","devicename") in line #260 of the LOAD program, and may be edited to suit your convenience. It overrides any name written into the Formatter program file on disk.

(ii) The Formatter will automatically display the filename last used by the Editor in Save File or Load File if the Formatter is loaded immediately after using the Editor.

(iii) The <fctn-9> key now returns the system to an option which allows return to the start of the Formatter with ENTER or the normal QUIT with the menu to be reloaded by selecting XB. This happens at the end of a formatting job as well.

(c) Utility

When the Utility option is selected the usual filename is presented for editing. Any legitimate disk file name may be entered here, with the disk drive specified as DSK1 or 2 or 3. The file so named must be consistent with E/A SAVE header protocol.

(i) TI-Writer normally hands over control in the GPL workspace, with a full set of characters loaded from GROM even if they are immediately overwritten by CHARA1. On occasion it may be desired to load programs which are designed for E/A conditions, in the absence of DSK1.CHARA1. These programs will usually assume Graphics mode, the workspace at >20BA, with sprites suppressed, the usual ASCII character patterns in place, color table loaded, and the rest of the pattern and sprite tables cleaned out. Such programs will not necessarily handle TI-Writer conditions gracefully.

(ii) A hook is provided in the XB LOAD program to allow choice between these two major alternatives. The statement in line #310 that loads the Utility program -- CALL LINK("UTILA",A$,B) has two parameters. A$ is the file-name and B is a numeric (logic valued) parameter. If B is zero the loader will assume Graphics mode and set up conditions for that, ignoring CHARA1 altogether, and if B is non-zero it will assume a TI-Writer compatible file and load DSK1.CHARA1, overwriting the XB normal size character set. The XB LOAD program can be edited to suit your particular application, without need to get into the Assembly coding. The parameters in this or any other CALL LINK must

by constants or simple variables.

(iii) If EDITA1 is loaded from the Utility option, SD will not function and the program will crash the machine on exit. FORMA1 will load but immediately bomb out. It is possible, but not likely, that a Utility program will overwrite the sprite loader and crash the computer. If it does you are out of luck. It is the responsibility of the Utility program to ensure return to the title screen on exit.

3. Programmer Notes

The source code for this program will be made available to anyone who can make good use of it. All that we ask as qualification is that you send some substantial material of yours in Assembler or Forth, preferably complete with source code and documentation, and that if you manage to make significant improvements to FUNLWRITER to let us know the details.

We have a firm belief that, as a result of TI's misbegotten marketing policies, the capabilities of the TI 99/4a have yet to be anywhere near fully explored, and that free exchange of programming techniques is the only way for users of this appealing orphan to get the best from their investment. We in Newcastle in the newly formed HV99 group are acutely conscious of the problems of lack of flow of information, both because of geographical isolation and because of the practice of TI-SHUG, the former affiliation of many HV99 members, of keeping much of the more interesting software that came their way within a close inner circle in Sydney, while making no noticeable contribution of any quality to the world of public domain software. We don't like having to re-invent the wheel. We do it if necessary, but it is a waste of time and effort better spent on forward progress.

The operation of the loader destroys the XB environment and it is necessary to return to the title screen each time. A Minimem version could mimic TI-Writer very closely with all the present enhancements as well, but defeats the intention of having TI-Writer accessible from the module usually resident in the console.

The Show Directory routine cannot be used as a free-standing program as it makes use of a number of the internal subroutines in the Editor.

The LOAD program was prepared by making a XB program file with enough free space at the top of high memory to imbed the machine code loader and directory files. The CALL LOADs in the XB listing are there to construct the REF/DEF table and update the LFALM pointer in low memory. The file was prepared by SAVEing the LOAD XB code in MERGE format. Then after NEWing the machine, enough dummy REM statements to give space for the machine code files were entered with line numbers below those of the original program, and the original program MERGEd back in. This makes sure that the relevant parts of the XB program are located in memory immediately adjacent to the line number table. Then the start of line number table address, validation word and end-of-file marker in the XB program file header were altered to trick XB into thinking that the line number table extends only to cover the original XB program. The machine code loader and directory files, each roughly 1.8K long, are next entered as absolute code in the area formerly occupied by the REM statements, and the file immediately SAVEd back to disk. It is the intention of F/Farm and HV99 to issue in the near future a further Public Domain disk with a pre-prepared template set of such programs with spaces for machine code, 1K to 10K, with detailed instructions for program development, and a sample program, such as an E/A object file loader which we have running in this format.

The XB program so created survives program editing and reSAVEing on this black, 1981 title screen console, but if difficulties are experienced let us know the details. The alternative is invocation of XB's CALL LOAD on an object file, a process almost as painful as watching a Commodore 64 disk drive. The LOAD program cannot be RESequenced without losing its integrity.

The Utility option basically assumes that the utility programs are free-standing, that is they contain all necessary system utilities such

as VMBW, DSRLNK, etc etc. The loader however leaves the Extended Basic utilities in place, and further adds a DSRLNK, none other in fact than the one from New Horizons User Group in Ohio by John Clulow. It came our way just as we were facing up to cobbling our own together from that marvellous mine of information, the TI-Forth source listing. Thanks John ! The entry address (BLWP) for this routine is >2532. As an example of how this may be used, a utility program file DPC is provided on the FUNLWRITER disk. This is just the TI disk fixer DPATCH as given to user groups by TI, but put in program file format with E/A utility calls converted to XB addresses.

July 1985
Tony and Will McGOVERN
215 Grinsell St.
Kotara, NSW 2288
AUSTRALIA

****
* PRIME NUMBER GENERATOR *
****

Here is a program for finding prime numbers that works. It runs in XB but is a bit slow finding the larger numbers but is O.K. for smaller numbers (<~50) It works this way (in thoery anyway, the program is a bit different), it takes a list of numbers then looks through and takes out any that are divisible by 2. After it has done this it repeats for 3,4,5 etc. until it reaches the end of the list. The numbers that are left are the primes.

C.MAC

```
100 ! SIEVE OF ERATOTHENES
110 ! C.MAC
120 ! JUNE '85
130 ! FOR H'V 99'ERS NEWS
140 CALL CLEAR
150 PRINT "SIEVE OF ERATOSTHENES-"
160 PRINT "FOR FINDING PRIME NUMBERS"
170 INPUT "STARTING AT ? ":0
180 INPUT "GOING TO ? ":A
190 DIM B(1000)
200 FOR S=2 TO A
210 B(S)=S
220 NEXT S
230 FOR K=2 TO A
240 FOR T=K TO A
250 IF B(T)/K=1 THEN 270
260 IF B(T)/K=INT(B(T)/K)THEN B(T)=0
270 NEXT T
280 NEXT K
290 FOR U=0 TO A
300 IF B(U)<>0 THEN PRINT B(U)
310 NEXT U
```

****
* JUNIOR SOFTWARE REVIEW *
****

TI-TREK

This game could for all it was worth have been called "IMPOSSIBLE MISSION". The game is based a lot on the random placings of the Klingons and your knowledge of angles and distances (and a lot of luck).

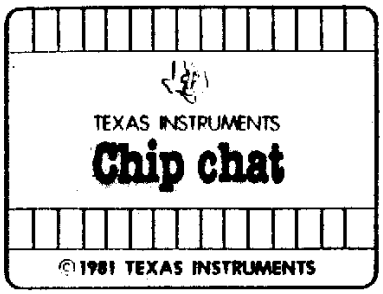
If you intend winning, well forget it. Your time is limited and enemies almost unlimited. To destroy the Klingons you can either move very close to them and fire phasers, I find this a waste of time and of power, it's not very effective and may not destroy the Klingons. The other method is to torpedo them which is where you need to know your angles; The computer will ask at what angle you will fire the torpedo, the Klingons just sit there as a torpedo crawls towards them; Torpedoes destroy anything they hit.

To get from one sector to another you must warp, if you try to warp great distances you may encounter a magnetic storm which will fling you off course to an unknown sector. If you are low on power or torpedoes you can find a mother ship to dock on by looking at your chart of the Galactic Sector which shows the number of stars, mother ships and enemies.

It is a very hard game to beat but a reasonably easy and enjoyable game to play if you have the instructions handy. The graphics are good and the sound rather interesting, although you would expect that from the original TI made game of STAR TREK.

PAUL SLOWEY.

NOTE: Junior members who would like to review software for inclusion in future issues of the magazine should get in contact with the club librarian ALAN LAWRENCE either at club meetings or by phone. Ph.486509



CALL KEY SUBROUTINES

When you specify a "key-unit" of 1 or 2(for split keyboard scan) in the CALL KEY subroutine, the computer does not return a true Ø(zero) in the return-variable when you press X on keyboard 1 or M on keyboard 2.
Therefore, if your programme checks to see whether the return-variable equals zero, the computer returns an answer as false.
The following example illustrates another way to write your program.

```
120 CALL KEY (1,A,B)
130 IF B<>1 THEN 100
140 IF A+1=1 THEN 200
```

Note that, if line 120 read "IF A=Ø THEN 200," the programme would not work properly.

CHESS MODULE BUG

A number of people who have the CHESS module have mentioned that the have been unable to utilise the "HELP" function. Fortunately this situation is easily remedied as it is merely a miss-print in the instruction manual. To utilise the Help function simply press FCTN< and you will have the "99/4A" checkmated before you know it!

SPRITES

Some TI LOGO and TI EXTENDED BASIC programs including sprites, one of the sprites may move into row 208, which is off the bottom of the display. If so, that sprite, and all sprites numbered greater than that one become invisible until the sprite in row 208 moves to another position. Thus, the sprites appear to "flicker" briefly.

MONITOR PIN-OUTS

There appears to be a great deal of differing opinions on the correct pin connections for a monochrome monitor. I have just completed the exercise and have found the following correct.

PIN 1. +12 VOLTS
PIN 2. COMPOSITE VIDEO
PIN 3. R-Y
PIN 4. B-Y
PIN 5. SOUND
PIN 6. EARTH

EXTENDED BASIC IS BACK

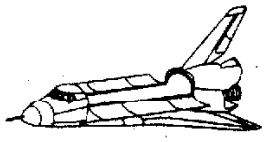
Under licence to Texas Instruments Inc. An American software company MICRO PAL has produced a new EXTENDED BASIC MODULE, which is 100% compatible with all programs written in TI EXTENDED BASIC.
All functions are identical with TI XB and the module comes complete with a 240 page manual.
With MICRO PAL Extended Basic you can run the hundreds of programs that require XB. Programs can automatically access the 32K memory expansion and run Auto-load disk based programs.
Best of all programmers are able to utilise all those extra features the TI99/4A is famed for, smooth sprite graphics and a 400 word built in vocabulary.

GOOD NEWS FOR ADVENTURERS

TEX COMP U.S.A. have announced a great new utility for adventure module owners. The all new program allows you to write your own adventures, or if you like edit, alter, list or copy to any medium (tape/disk) any existing adventure compatible with the Adventure module. You can now freely create Adventure games using a "template" game as a start up step. These games can be conceived using the full power and capabilities of the TI Adventure module.
The programme is written for the MINI-MEMORY or EDITOR/ASSEMBLER modules and offers a full screen editor and a special easy to use mnemomic language called APL (Adventure Program Language) and comes complete with a 75 page manual.

HUNTER VALLEY 99'ERS

LIBRARY NEWS



HI 99'ers,

More musings from the dungeon by the Bay. Well if you are still having problems loading from cassette it may be possible that the HEAD is not lined up to the same level as the orignal recording HEAD .We are trying to get special gear to line up all heads to the same standard. But in the meantime, I have read a TIP that may help: the computer is looking for a signal of 1 volt measured peak to peak. To obtain this it is necessary to construct a ' Y ' connector to allow the conection of a Volt-Ohm-Meter to be connected to the ear phone jack from the tape recorder. Next set the V.O.M. to 2.5 volts A.C. and adjust Vol(while loading the programe)To give approximately 1 volt.Leave Tone at maximum if fitted and only adjust tone if all else fails. I have tried this on some difficult programs and found it sucessful (source MICROpendium from a letter by W.Fielden) At the next few meetings I will bring a demagnetiser and V.O.M. meter and 'Y' conector I made up plus other gear if we can borrow it to line up the heads.

The BASIC Handbook

David A.Lien Encyclopedia of the BASIC Computer Language covers most BASIC Words used by just about every compurer,cira 1981.The author Lien does not set out to compare computers or advantages dissadvantages of each but gives short routines or test programes to find out if it will run on your Computer,also variations in usage showing how ALL computers capabilities can be enhanced by a good grasp of BASIC when converting programs from one machine to another. This would by a good book for pupils using other computers outside the I envirоment. Machines covered include APPLE , TRS-80 , ATARI , SINCLAIR , PET , AND our own beloved -TI/99.I have spotted this book at the UNI.BOOKSHOP,DICK SMITHS,and at A.R ROBERTSON$ for ABOUT $23.00.

This is being compiled via FUNNEL WRITER ver 2.0 and I have been informed of a further mod! It is hard to comprehnd how one can improve further on a good basic product but our intrepid TONY with help from son WILL McGOVERN sure has done just that! Not only can you have screen colours of your choice a'la orignal but S.D shows directory without crashing programe to test pattern as was a problem with some old XB loaders. In addition one can load utility programes with a loader that "moves" so once again no more 'musicial graphic displays'as your 99 goes walkabout!! as well you can default to your printer by changing lines 250 , 260. Also using option 3 you can load and run any program in machine code PROGRAM format. Full documentation is included on the disk. Any body out there want to buy a pre-loved TI.Writer only used once on Sunday.
P.S. I wish it could type, Maybe thats next!!!

Well good news for those with only the basic cassette system one of our members BRIAN RUTHERFORD has done a magnificient Word processor in XBasic in last months mag, and to follow up this month he has a formatter for you to type in. Both programmes are fully compatiable with Funnel Writer format.So there is no excuse for you not submitting articles to YOU OWN NEWSLETTER. Remember it does not have to be a programme or technical in content.

I have been testing out a disk lister/cataloger by JOE WRIGHT and have the library listed yet again !! The beauty of this programme is the ability to DELETE files or disks as required,and I no longer can grab a cuppa while the lists are being sorted.Watch out for this routine in the future SORTING ARTICLES. If anyone wants copies of any of the above let us know by phoning or drop a line to the HV99'ers to let me know.

Adventurers

Any keen people who would be interested in starting an adventure group meeting along lines of Dungeons and Dragons.Perhaps it could meet same KNIGHTS as the regular basic classes.

Thats it for now,

Happy programing,

Al Lawrence.

Last word

As the title suggests this page will be reserved for any last, last minute news before the magazine goes to the photocopier. If you would like anything included on this page please get in contact with a committee member and give him all the relevent details.

HARDWARE WORKSHOP.
At the last committee meeting it was proposed that a hardware workshop be organised. The aim being that interested members could bring along their consoles for cleaning, fitting of reset buttons, and for anyone interested the fitting of 32K of extra memory.
Anyone interested in having any of the above would be advised to contact AL LAWRENCE and let him know what you require. The only expense incurred will be for parts,consumables and band-aids.
Perhaps amongst our members there is a hardy soul who would like to take up the leadership of this project?? With the view in mind of possibly eventually forming a sub-group interested in designing and constructing hardware add-ons for the TI99/4A.

JOHN PATON of Rutherford requires a RS232 Card for the PE Box. Anyone who can help can contact John at PH. 049-326014

PETE SMITH has MULTIPLAN for sale for $80.00 Ph.049-336164

LOAD INTERRUPT BUTTON. Anyone who has any information regarding the installation of a load interrupt button could you please pass it along to us as soon as possible as we would like to be able to offer this modification at the HARDWARE WORKSHOP.

URGENTLY REQUIRED any information on the AXIOM Interface. Contact Peter Coxon. PH. 049-751930

WANTED EXTENDED BASIC. Phone Master PAUL SLOWEY PH.049-594279

PLATO COURSE DESIGNER AUTHORING SYTEM. Maybe our American readers could help us with this request. If you have the program or any information regarding it, could you please pass it on to us care of the Editor HV99'ers.

NEIL QUIGG our intrepid hardware hacker has completed building a 32K expansion card for the PE BOX. Neil has based his design on information received from TIUP. It is currently still under trial but at this stage results look great. Thank you BERNIE and PHIL from TIUP. NEIL has also converted his B&W television into a monitor for his TI. Both these projects will be presented as articles in future edition of the magazine.

WANTED Public Domain TERMINAL EMULATOR Programmes in exchange for other interesting Public Domain software. We would also be interested in any information that would assist us in the construction of a modem suitable for use either in the PE BOX or stand alone.

Steve Taylor