Oct 85

HUNTER VALLEY 99'ERS NEWS

TI 99/4A

HOME COMPUTER NEWSLETTER

NEWSLETTER No. 5

TEXAS INSTRUMENTS

Newcastle & The Hunter Region

TI-99/4A

Home Computer USERS' GROUP

DISCLAIMER

The HV99 NEWS is the official newsletter of the HUNTER VALLEY NINETY NINE 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 NEWS as a result of applying such information.
TEXAS INSTRUMENTS trademarks, names and logos are all copyright to TEXAS INSTRUMENTS.
HV99 is a non profit group of TI99/4A computer users, not affiliated in any way with TEXAS INSTRUMENTS.

CONTRIBUTIONS

Members and non members are invited to contribute articles for publication in HV99 NEWS.
Any copy intended for publication may be typed, hand written, or submited 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 program will be supplied if required by the club librarian Al Lawrence.
Please include along with your article 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, right justified.
All articles printed in HV99 NEWS (unless notified otherwise) are considered to be PUBLIC DOMAIN. Other user groups wishing to reproduce material from HV99 NEWS may feel free to do so as long as the source and author are recognised.
Articles for publication can be submitted to.

THE EDITOR
HV99 NEWS
15 GAYTON CLOSE
WARNERS BAY 2282
NEWCASTLE

General address for ALL other club related correspondence.

THE SECRETARY
HV99 USER GROUP
25 RESERVE RD.
WANGI 2267
NEWCASTLE

YOUR COMMITTEE 1985

A. WRIGHT	PRES.	PH. 488120
P. COXON	SECT.	PH. 751830
B. RUTHERFORD	TRES.	PH. 488184
A. LAWRENCE	LIBR.	PH. 488508
S. TAYLOR	EDIT.	PH. 487078
B. WOODS	EDAS.	PH. 588307
T. MC. GOVERN	TECH.	PH. 323188
B. MAC. CLURE	MODL.	PH. 437431
G. JONES	TECH.	PH. 573744
D. WINTON		PH. 581933

SECRETARY'S NOTES

Hi once again!

It seems only five minutes since I was writing last months notes!. Well the postal situation has put all our Overseas mail into the hold position so unfortunately there is very little news in that department.

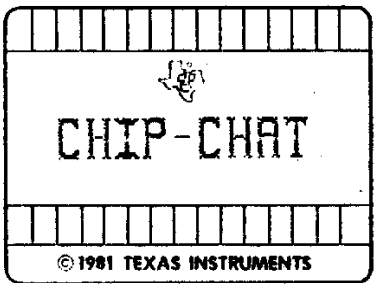
The matchbox 32K should be going into the machines of those members who ordered them by the time this issue goes to press. We have two or three blank boards available for the cost of production and mailing if anyone is interested...Many thanks to those who gave their time to help on our solder night.

The Club's printer has been ordered and should be available by the end of November. The delay was caused by our first supplier running out of stock. We have ordered the printer from Microbee, Newcastle and I think you will all be pleased with the product.

To all our expert programmer's, and those not so expert! remember our bumper issue is coming up in December and all contributions will be greatfully accepted! Could you arrange to have them to theEditor by 30th November (no later). Remember there is no January issue or January meeting because of the Christmas hols. Things will roll again on 11Th February 1986.

Have you all got your orders in to Santa? Why not see Al Lawrence for some good ideas.
HAPPY COMPUTING

PETER C.



* TI PUBLIC DOMAIN SOFTWARE *

Owners, users and programmers of the TI99/4A can now market their own software programs in a software newsletter, SPM, devoted to expanding the usefulness of Texas Instruments computers. Advertising is free for these people who subscribe for $US 18.00 a year. For an information package, send $US2.00 which will be deducted from the subscription fee. Contact R. Clark, SPM, RD#4, box 90-A, Bath, NY 14810 U.S.A.

* TI99/4A BITS FOR SALE *

Anyone wishing to purchase a spare keyboard or regulated power supply for their computer should contact DO KAY COMPUTER PRODUCTS of 2100 De La Cruz Blvd. Santa Clara, CA 95050. Keyboards sell for $US 6.95 and the power boards are the same.

* DataBioTics RELEASE NEW SOFTWARE *

Californian software company DataBioTics has just released four new TI products.

Probably the most interesting are MINI WRITER II AND III both of which are cartridge based wordprocessors with Mini Writer III having a special interface built into the back of the module so that it can simply be plugged straight into a printer!!.

Both versions offer full screen editing, 12K text buffer, Disk and cassette compatibility, right justification, word wrap and variable line length.

Mini-Writer III sells in the US for $US 99.95.

Also released are a Disk Master I, a Corcomp workalike and Pilot.

The new version of Pilot contains only eight commands and is available for $US 24.95.

For further information contact DataBioTics, P.O. Box 1194, Palos Verdes Estates, CA 90274. U.S.A.

* THE FORTH DIMENSION *

FORTH DIMENSIONS is a 44 page Bi-Monthly magazine produced by the Forth Interest Group. The three issues I have had an opportunity to read mainly cater to the more advanced user but a look at the Volume 6 index shows some TI99/4A support.

The Forth Interest Group is a worldwide non-profit member supported organisation with over 5000 members and 80 chapters. A FIG membership includes a subscription to FORTH DIMENSIONS.

FIG also offer its members publication discounts, an on-line database, a job registry, a large selection of Forth literature, and many other services. Cost via surface mail is $Aust 27.00 or air $Aust 33.00.

The annual membership dues are based on the membership year, which runs from May 1 to April 30. When you join, you will receive issues that have already been circulated for the current volume of FORTH DIMENSIONS and subsequent issues will be mailed as published.

* NEW MILLERS GRAPHICS CATALOG *

Al Lawrence has just received a copy of the Summer Millers Graphics catalog. Some of the new software releases include "EXPLORER", a program that will allow poeple with thefull system to take over control of the computer!!. "ADVANCED DIAGNOSTICS" another disk based utility which reveals how information is stored on a disk as well as providing several other useful tasks and "NIGHT MISSION" a cassette based Extended Basic game featuring five challenging screens and impressive graphics. This program also comes with a book which picks up where the Smart Programmers Guide for Sprites left off by fully documenting each line of program flow. It is complete with screen diagrams, character lists with diagrams, and variable lists. Another chapter in the book explains the use of the powerful AND function, through example listings, to help speed up your programs and save bytes. There is also a section on CALL PEEK and CALL LOAD that lists many various locations in the computer and documents their use.

TOKENISED COMMANDS IN XB

THE TI99/4A represents its BASIC commands internally in a tokenised, or numerically abbreviated form. By trial and error it has been found that when using Extended Basic that certain keystrokes also generate the same codes as these tokenised commands.

Due to the limitation of being only able to enter one command per line plus the absence of any documentation on the subject in any manuals the impression is given that this is only any idiosyncrasy of the system and not an intentional design feature.

To take advantage of this hidden feature you must depress the CTRL key and then the key which represents the hidden statement.

For example if you are keying in a program that contains a lot of "PRINT" statements simply type CTRL-:. Nothing will appear on the screen but when you list your program PRINT will be there.

Although TI states that it was not their original intention for these functions, and that they do not recommend this type of programming method, I know several people using it and they find that it works quite well.

The hidden statements and the keys that make them operate are listed below.

1(TO)	U(RANDOMIZE)
5(:)	6())
9(OPEN)	Ø(THEN)
W(READ)	E(GO)
Y(DELETE)	Z(STEP)
P(TRACE)	\(AND)
D(IF)	F(GOTO)
J(DIM)	K(END)
K(REM)	X(STOP)
B(::)	N(BREAK)
3(,)	4(;)
7(()	8(OPTION)
=(CALL)	Q(UNTRACE)
R(INPUT)	T(RESTORE)
I(DEF)	O(UNBREAK)
A(ELSE)	S(DATA)
G(GOSUB)	H(RETURN)
L(FOR)	;(PRINT)
C(!)	V(NEXT)
M(LET)	>(ON)

Try out these timesavers then experiment yourself, if you find any more please pass them on to us so that they can be printed in a future issue of HUG99 NEWS.

CorComp NEWS

CorComp are still coming up with new ideas for the 99/4A according to the latest CorComp Cursor Newsletter. The company has announced no less than five new products for the 99/4A. CorComp has also joined with Quality 99 Software to produce the exciting range of new software. With the two companies working together they claim to be able to make the most use of the best features of the 4A.

The five new products are: A GROM buster that allows 1983 V2.Ø consoles to run Atari software written for the 99/4A. A load interrupt switch which allows the use of very quick screen dumps. A stand alone 32K memory. A 9900 clock calendar which is as said,a clock with battery back up and a calender which can be called from Extended Basic or Basic with one simple command, features are year, month, date, day, hours, minutes and seconds also included with the clock calender is a load interrupt for screen dumping.

The final product mentioned for release was a new card for the expansion box named the Triple Tech Card, this card has the features of the above clock calendar stand alone unit, plus a port where you can insert the card from your speech synthesizer onto the Triple Tech card and have speech inside your expansion box without loss of any of the speech synthesizers features. The other part of the Triple Techs features that sounds great to me is the 64K print buffer that can connect to your parallel port on the RS 232 card and allow a full 64K of information to flow to the buffer for printing at the speed the printer can handle and this in turn allows the computer to be performing other tasks, anyone with a daisy wheel printer will welcome this new card just for the convenience of this alone, but be warned, we are led to believe that this printer buffer does not work on the serial port of the RS 232 card, only the parallel.

COURTESY of Rex Shephard Tasmania.

CHIP-CHAT INPUT

If you have any newsworthy items that you feel could be used in the Chip-chat column I would appreciate it if you would pass them on to me.

ED.

TRANSLITERATION AND TI-WRITER

BY
BRIAN WOODS HV8B

The Transliterate command (.TL) allows you to combine multi-character printer control codes into a single character for the Text Formatter to read. It allows you to alter print styles to add emphasis etc. to your printout.

Transliterating the various control codes to a single character and using TI-WRITER's Special Character Mode embeds a non-printed character in your text (in Text Editor mode) that appears on screen but not in the printout. When the Formatter encounters this character during printing it is recognized as a print command and acts accordingly.

I have used the Transliterate command to allow all of the print types available on an Amust-100 printer to be accessed in TI-WRITER simply by using CTRL U, a letter, CTRL U.

Below is a list of the TL commands I have used:

```
EMPHASISED ON: .TL 17:27,69
EMPHASISED OFF: .TL 23: 27,70
ITALICS ON: .TL 5:27,52,1?
ITALICS OFF: .TL 18:27,52,0
CONTIN U/LINE ON: .TL 20:27,45,49
CONTIN U/LINE OFF: .TL 25:27,45,48
CONDENSED ON: .TL 1:15
CONDENSED OFF: .TL 19:18
DBLE WIDTH ON: .TL 4:27,87,49
DBLE WIDTH OFF: .TL 8:27,87,48
DOUBLE STRIKE ON: .TL 7:27,71
DBLE STRIKE OFF: .TL 8:27,72
SUPERSCRIPT ON: .TL 26:27,83,49
SUPER/SUBSCRIPT OFF: .TL 24:27,84
SUBSCRIPT ON: .TL 3:27,83,49
ENLARGED ON: .TL 22:27,14
ENLARGED OFF: .TL 2:27,20
```

Now when I am typing in a document using the Text Editor the first line is ".IF DSK1.CODE2", which is the filename of my TL commands, then when I need to use them I go into Special Character mode (CTRL U), type the letter that refers to my requirements, then CTRL U again. It's as easy as that.

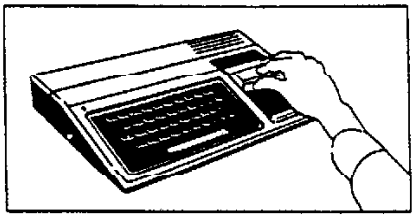
Below is a list of the type fonts and the letter required to access them, based on the transliterated commands above.

TYPES	IN	OUT
EMPHASISED	@	W
ITALICS	E	R
CONT U/LINE	T	Y
CONDENSED	A	S
WIDE	D	F
DBLE STRIKE	G	H
SUPERSCRIPT	Z	X
SUBSCRIPT	C	X
ENLARGED	V	B

The difference between wide print and enlarged is that enlarged print will only print in that mode to the end of the line unless turned off before, and wide print remains in that mode till turned off.

When using these codes, the symbol appearing on the screen is not printed, but remember to leave a space between the symbol and the next letter of text otherwise the two words will be joined.

By using these commands it has made it easier to utilise the printer's capabilities and add a little interest to my printouts.



TEXAS ORPHANS

The 28th. of October 1985 marked the second anniversary of Texas Instruments withdrawal from the "home computer" marketplace. Since the demise of the TI99/4A several other computer manufacturers have followed TI's course; the most recent being IBM's withdrawal of the unsuccessful PC Jnr.

For a lot of people who purchased their TI99/4A during the ensuing "sell out" period the time has now come to decide whether to stick with their purchase and expand their system or move to another brand with a broader software base.

I'm sure that already unfortunately a large number of 99/4A's have taken up residence collecting dust on top of wardrobes or have been virtually given away in the Saturday classifieds or Trading Post.

Those who have made the decision to continue on with their 99/4A and resist the temptation to change brands have been rewarded by what can only be described as a flood of new hardware and software support, with no signs of this diminishing in the foreseeable future.

It now appears that the prophets of doom who predicted an early demise for our machine were wrong. TI's exit has in fact provided the incentive for people to delve into the architecture of the machine in order to fully realise its potential; their new found knowledge being passed on in the form of vastly superior quality software.

Undoubtedly TI's exit has prompted this phenomenon; as TI made no bones of the fact that they did not want outside programmers to write software for the TI994/A. Software being the most profitable part of the computer business; a $40.00 programme cost TI around $6.00 to produce. TI therefore hated the idea of sharing this incredible profit with anyone. So instead of making it easy for third party software companies to design and market software for the 99/4A TI went deliberately out of its way to make things difficult, even to the extent of making internal adjustments to the machine that kept outsiders from writing software for it.

The end result of this shortsightedness is only too obvious by the fact TI99/4A owners have only a small number of commercially produced software packages from which to choose, whereas the owners of Apples and Commodores have thousands. Furthermore people in a position to distribute the remaining TI inventory in Australia have greatly inflated the prices of useful application packages such as TI Writer, Editor Assembler etc etc to the extent that it is now far cheaper to fully import the items, the cost still being less than the local price.

So what does the future hold? If you are reading this column then you can consider yourself a dyed in the wool 99/4A Nut!! and the only way you are going to survive is to remain in a user group; and this brings me to the HUNTER VALLEY NINETY NINE USER GROUP for I feel it is time to take a quick look back over what has happened in the six short months of our existence.

I personally feel that significant gains have been made in establishing a user group which caters to the needs of the majority of its members. Any doubts as to the group success have been well and truely squashed by the enthusiastic reesponse to the questionnaire and subsequent recruiting campaign.

The aims of the group now are to pool the experience and resources of all the members so that everyone learns and benefits. To this end regular classes in basic, XB and Forth are already established with people like Garry Jones, Tony Mc. Govern, Richard Terry and Joe wright sharing their hard won programming knowledge. Behind the scenes the committee is working to shape and guide the group on a long term basis, we are not just looking towards next year but five or even ten years down the line. To achieve our goals we need your help, the group just wont survive unless we all contribute. If you aren't happy with the direction the group is taking let the committee know. One thing to remember though,the group recquires more than criticism to survive; it needs your support.
ED.

MINI WORD PROCESSOR DISSECTED PART 2

To continue on from last month we will take off from line 210 CALL CL(L$(),L)::RETURN. CL being short for change line, with the CALL statement passing the array L$, which if you all remember is the array we store the text that has been typed in, also the variable L which is the number of text lines that have been LINPUT so far.

We find the subprogramme at line 510 where the array is known as L$ again and the variable L from the main programme is named L again. The screen is cleared and the subprogramme LI is called, and LI is short for line input. The value 23 and 3 plus the string "Which line" and a variable named D and our old favourite L are passed to the subprogramme LI.

We find the subprogramme LI back at line 390, where the first two values are transfered two the variables A and B respectivly, and the string is passed to the variable A$. Again just to confuse you I have transfered the variables D and L to variables D and L. A$ is then displayed at row A column B, and by doing things this way we can display what ever message we want were we want it any time we call the subprogramme. Next at line 400 ACCEPT AT(A,19)VALIDATE(DIGIT)BEEP:A$::IF A$="" THEN 400 ELSE D=VAL(A$). Again using the variable A for the row and then to save some memory we use A$ to to store the input, which is validated to accept a digit only. Next the input is tested to see if it is a null string if so we just loop back to have another try. The reason I used a string for a numeric input is because even though the input is validated for a digit only, it is still possible for a user to just press ENTER and enter a null string, and if that happens when the input needed is a numeric variable an error message will be displayed, which messes up any screen completely. D is then set to the VAL of A$ and the control passes to the next line. Line 410 where D is tested to make sure it is not less than one or greater than L, not forgetting that the value stored in L is the highest text line that has been input. If D is with in the required parameters the control passes to line 420 the subend. But if it is not with in the required parameters a warning message is displayed and the subprogramme KC is called. KC looks after all the "PRESS ANY KEY TO CONTINUE" situations and it is one subprogramme that does not need explaining. After the subprogramme KC has been completed and control has returned to LI on line 410 D is set to zero and the subend performed, with control returning to CL and line 510.

Back at line 510 where D is tested to see if it equal to zero and if so the subexit is performed back to the main programme. That is one of the "outs" I gave the user if they realised they had pressed a wrong key some where. Then at the next line number 520 the first thing we do is to call another subprogramme, this one is called LC for line check. Where we pass the array known as L$ and two variables D and K and D is the number of the line of text we wish to change.

SUB LC starts at line 670 where the array is again known as L$ and the variables are known as D and K again. Next the line of text L$(D) is displayed with the message, "This line..... Y/N". At the next line 680 a simple call key "D" is performed and if neither a N or Y key is pressed the programme just loops back. As soon as a correct key is pressed the subend at line 690 is performed, taking the ASCII value of the key that was pressed in the variable K, to the variable K in SUB CL.

Again back to CL and the continuation of line 520 where if K=78, that is N was pressed because it was not the line you wanted to change, the SUBEXIT is performed back to the main programme and the RETURN at line 210. If N was not the key pressed the programme continue's on with line 530. Where the length of the text line L$(D) is divided by 28, the length of a screen line to find out how many screen lines long it is, and the result stored in the variable LE. If LE is not a whole number then LE is made equal to "INT" LE and 1 is

added to it. Next at line 540 a dummy CALL KEY "5" is performed to turn the full keyboard back on and the text line you are about to change is displayed in the middle of the screen. A variable P is given the value of 1 and string variable A$ is set to "" a null string. Next at line 550 a loop is set up with the limit being LE, the number of screen lines long the text is. The text is then displayed one screen line at a time with the statements P$=SEG$(L$(D),P,28)::DISPLAY AT 24,1):P$. Then next statement is an ACCEPT without the AT and again the variable P$ is used for that input. Next A$ is made equal to A$ and P$, P is incremented by 28 ready to read the next segment of the text and then loops back for as many times as there are screen lines in the text. Then at 570 the length of A$ is tested to see if it is longer than 191 (191 bytes plus one byte for the length of the string is the maximum length file that can be saved using a cassette). If it is longer than 191 bytes an error message is displayed SUB KC is called and the programme loops back for you to have another go. If it was not to long then at line 580 the changed line is displayed and a message asking if the changes are alright. On then to line 590 where a CALL KEY "3" is performed so you only need a single key entry for acceptance of the Y or N and if it was Y then the length of the old line is added to the number of bytes free in memory, the old line is then made equal to the new line and the length of that is subtracted from the amount of bytes free in memory. P$ and A$ are reset to a null string to conserve memory and the subexit is performed back to the main programme. If it was any key other than a Y pressed the programme went straight to line 600 and if it wasent an N that was pressed it loops back to wait for another key press and if it was the N it goes back to line 540 for you to have another go at changing the line. Line 610 the programme never gets to but it is need to tell the computer where the subprogramme ends.

I think that will do for this month, as I find this explaining business harder than writing the programme. But if you have any questions about it dont be afraid to ask.

BRIAN.R HV99

MINI-MEM BATTERY !!

Apparently some people have checked with TI and found that it would cost heaps to replace their Mini Memory battery. However for those brave souls who are willing to replace the battery themselves, it can be done for $ 2.69. To find if your battery needs to be replaced, measure the battery voltage, it should be 3 volts, if it's much less than that, replace it with a TANDY RADIO SHACH CR2032 (CAT 23-162) Lithium calculator battery. These cells have a shelf life of between 5 to 10 years and should last almost that long in circuit. The case is the positive terminal just like the original but unlike it the CR2032 doesn't have leads and must be soldered on.

WARNING!!! Lithium batteries can be destroyed by heating them and certain types can explode !!!

Don't try to make this modification if you don't think you are competent, you might destroy your Mini Mem, or worse.

Scrape the center of the case where you are to solder a 20 gauge wire. A lead from a 1 or 2 watt resistor is ideal. Melt a small glob of solder onto the end of the wire and quickly solder it to the battery case. This is best done with a 100 watt soldering gun. Be sure the gun's hot before you try to solder the wire on. Soldering should only take 1 second. Have a wet paper towel ready to press on the battery as soon as you remove the soldering gun. Insulation between terminals may be thermal plastic and could deform allowing the battery to short if you aren't quick. Cut the soldered lead close to the resistor body and flip the battery over and solder the lead on the other side, making sure that it doesn't touch the positive terminal. Be sure that this lead points 180 degrees away from the other lead so the battery will mount te same way as the original one. Bend the leads so they will fit into the slots for the original battery.

Before you remove the original, note that the positive lead is connected toward the outside of the board. Quickly solder the replacement in the same way. Check the voltage across the battery if it reads 3 volts, you're all set.

Richard J. Bailey
NH99ER USER GROUP.

CHRISTMAS MUSIC PROGRAMME PART 1

BY GARRY JONES HV99

THIS PROGRAMME WILL BE PRESENTED IN TWO PARTS TO GIVE YOU A CHANCE TO HAVE IT COMPLETELY KEYED IN BEFORE CHRISTMAS. PART ONE IS THE MUSIC SECTION AND WILL RUN BY ITSELF. THE GRAPHICS WILL BE IN NEXT MONTHS ISSUE AND IT WILL BE SIMPLY A MATTER OF MERGING PART1 AND 2

```
10 !****CHRISTMAS MUSIC****
20 !*******PROGRAMME*******
30 !**HUNTER VALLEY 99ER'S*
40 !*****USER GROUP********
50 !*********BY***********
60 !*****GARY JONES********
70 !*****NEWCASTLE*********
160 CALL CLEAR
210 DISPLAY AT(5,4)SIZE(23):
"1-RUDOLPH THE RED NOSED"
215 DISPLAY AT(6,12)SIZE(8):
"REINDEER"
220 DISPLAY AT(8,4)SIZE(16):
"2-THE FIRST NOEL"
230 DISPLAY AT(10,4)SIZE(14)
:"3-JINGLE BELLS"
240 DISPLAY AT(13,5)SIZE(13)
:"SELECT TUNE ?" :: ACCEPT A
T(13,17)SIZE(-1):T$ :: IF (T
$<"1")OR(T$)"3")OR(T$="")THE
N 240
250 CALL CLEAR :: ON VAL(T$)
GOSUB 3000,4000,5000 :: GOTO
 160
3000 M=325 !RUDOLPH
3010 RESTORE 3400 :: GOSUB 6
000 :: X=7 :: RESTORE 3600 :
: GOSUB 7000
3020 RESTORE 3410 :: GOSUB 6
000 :: X=7 :: RESTORE 3610 :
: GOSUB 7000
3030 RESTORE 3420 :: GOSUB 6
000 :: X=7 :: RESTORE 3620 :
: GOSUB 7000
3040 RESTORE 3430 :: GOSUB 6
000 :: X=7 :: RESTORE 3630 :
: GOSUB 7000
3050 RESTORE 3440 :: GOSUB 6
000 :: X=7 :: RESTORE 3640 :
: GOSUB 7000
3060 RESTORE 3450 :: GOSUB 6
000 :: X=7 :: RESTORE 3650 :
: GOSUB 7000
3070 RESTORE 3460 :: GOSUB 6
000 :: X=7 :: RESTORE 3660 :
: GOSUB 7000
3080 RESTORE 3470 :: GOSUB 6
000 :: X=7 :: RESTORE 3670 :
: GOSUB 7000
3090 RESTORE 3480 :: GOSUB 6
000 :: X=7 :: RESTORE 3680 :
: GOSUB 7000
3100 RESTORE 3490 :: GOSUB 6
000 :: X=5 :: RESTORE 3690 :
: GOSUB 7000
3110 RESTORE 3500 :: GOSUB 6
000 :: X=7 :: RESTORE 3700 :
: GOSUB 7000 :: CALL CLEAR
3120 RESTORE 3510 :: GOSUB 6
000 :: X=7 :: RESTORE 3710 :
: GOSUB 7000
3130 RESTORE 3520 :: GOSUB 6
000 :: X=7 :: RESTORE 3720 :
: GOSUB 7000
3140 RESTORE 3530 :: GOSUB 6
000 :: X=7 :: RESTORE 3730 :
: GOSUB 7000
3150 RESTORE 3540 :: GOSUB 6
000 :: X=7 :: RESTORE 3740 :
: GOSUB 7000
3160 RESTORE 3550 :: GOSUB 6
000 :: X=8 :: RESTORE 3750 :
: GOSUB 7000 :: RETURN
3400 DATA RU-DOLPH THE RED N
OSED       REIN-DEER,0
```

```
3410 DATA HAD A VER-Y CHIN-Y
 NOSE,3
3420 DATA AND IF YOU EV-ER S
AW IT,2
3430 DATA YOU WOULD E-VEN SA
Y IT GLOWS,2
3440 DATA ALL OF THE OTH-ER
REIN-DEER,2
3450 DATA USED TO LAUGH AND
CALL        HIM NAMES,2
3460 DATA THEY NEVER LET POO
R RU-DOLPH,3
3470 DATA JOIN IN AN-Y REIN-
DEER GAMES,2
3480 DATA THEN ONE FOGGY CHR
ISTMAS EVE,2
3490 DATA SAN-TA CAME TO SAY
,2
3500 DATA RU-DOLPH WITH YOUR
 NOSE SO    BRIGHT,2
3510 DATA WON(T YOU GUIDE MY
 SLEIGH     TO-NIGHT?,3
3520 DATA THEN HOW THE REIN-
DEER        LOVED HIM,3
3530 DATA AS THEY SHOUT-ED O
UT WITH     GLEE,3
3540 DATA RU-DOLPH THE RED-N
OSED        REIN-DEER,3
3550 DATA YOU(LL GO DOWN IN
HIS-TO-RY.,3
3600 DATA .75,196,196,.8,220
,220,1,196,196,1,165,165,1,2
62,262,1,220,220,3,196,196
3610 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,262,330,4,247,294
3620 DATA .65,175,220,.5,196
,247,1,175,220,1,147,175,1,2
47,294,1,220,262,3,196,247
3630 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,220,262,4,165,196
3640 DATA .65,196,247,.5,220
,262,1,196,247,1,165,196,1,2
62,330,1,220,262,3,196,247
3650 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,262,330,4,247,294
3660 DATA .65,175,220,.5,196
,247,1,175,220,1,147,175,1,2
47,294,1,220,262,3,196,247
3670 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,294,349,4,262,330
3680 DATA 1,220,262,1,220,26
2,1,262,330,1,220,262,1,196,
247,1,165,196,2,196,247
3690 DATA 1,175,22?,1,220,26
2,1,196,247,1,175,220,4,165,
196
3700 DATA 1,147,175,1,165,19
6,1,196,247,1,220,262,1,247,
294,1,247,294,2,247,294
3710 DATA 1,262,330,1,262,33
0,1,247,294,1,220,262,1,196,
247,1,175,220,2,147,175
3720 DATA .65,196,247,.5,220
,262,1,196,247,1,165,196,1,2
62,330,1,220,262,3,196,247
3730 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,262,330,4,247,294
3740 DATA .75,175,220,.7,220
,262,1,175,220,1,147,175,1,2
47,294,1,220,262,3,196,247
3750 DATA .5,196,247,.5,220,
262,.5,196,247,.5,220,262,1,
196,247,1,294,349,3,262,330,
1,20000,20000
4000 M=450 !FIRST NOEL
4010 RESTORE 4400 :: GOSUB 6
000 :: X=7 :: RESTORE 4600 :
: GOSUB 7000
4020 RESTORE 4410 :: GOSUB 6
000 :: X=6 :: RESTORE 4610 :
: GOSUB 7000
4030 RESTORE 4420 :: GOSUB 6
000 :: X=7 :: RESTORE 4620 :
: GOSUB 7000
4040 RESTORE 4430 :: GOSUB 6
000 :: X=5 :: RESTORE 4630 :
: GOSUB 7000
4050 RESTORE 4440 :: GOSUB 6
000 :: X=7 :: RESTORE 4600 :
: GOSUB 7000
4060 RESTORE 4450 :: GOSUB 6
000 :: X=6 :: RESTORE 4610 :
: GOSUB 7000
4070 RESTORE 4460 :: GOSUB 6
000 :: X=7 :: RESTORE 4620 :
: GOSUB 7000
4080 RESTORE 4470 :: GOSUB 6
000 :: X=5 :: RESTORE 4630 :
: GOSUB 7000
4090 RESTORE 4480 :: GOSUB 6
000 :: X=7 :: RESTORE 4600 :
: GOSUB 7000
```

```
4100 RESTORE 4490 :: GOSUB 6
000 :: X=5 :: RESTORE 4640 :
: GOSUB 7000
4110 RESTORE 4500 :: GOSUB 6
000 :: X=5 :: RESTORE 4650 :
: GOSUB 7000
4120 RESTORE 4510 :: GOSUB 6
000 :: X=5 :: RESTORE 4660 :
: GOSUB 7000 :: RETURN
4400 DATA THE__ FIRST__ NO-E
L,0
4410 DATA THE__ AN-GEL DID S
AY,2
4420 DATA WAS TO CER-TAIN PO
OR          SHEP-HEARDS,2
4430 DATA IN FIELDS AS THEY
LAY;,3
4440 DATA IN__ FIELDS__ WHER
E__ THEY,2
4450 DATA LAY__ KEEP-ING THE
IR SHEEP,2
4460 DATA ON A COLD WIN-TER(
S NIGHT__,2
4470 DATA THAT WAS__ SO DEEP
,2
4480 DATA NO-EL__ NO-EL NO-E
L,2
4490 DATA NO-EL NO-EL,2
4500 DATA BORN IS THE KING__
,2
4510 DATA OF IS-RA-EL.,2
4600 DATA .5,165,165,.5,147,
147,1,131,131,.5,147,147,.5,
165,165,.5,175,175,2,196,196
4610 DATA .5,220,262,.5,247,
294,1,262,330,1,247,294,1,22
0,262,2,196,247
4620 DATA .5,220,262,.5,247,
294,1,262,330,1,247,294,1,22
0,262,1,196,247,1.5,220,262
4630 DATA 1,247,294,1,262,33
0,1,196,247,1,175,220,2,165,
196
4640 DATA .5,262,330,.5,247,
294,2,220,262,1,220,262,3,19
6,247
4650 DATA 1,262,330,1,247,29
4,1,220,262,1,196,247,1.5,22
0,262
4660 DATA 1,247,294,1,262,33
0,1,196,247,1,175,220,2,165,
196
5000 M=300 !JINGLE BELLS
5010 RESTORE 5400 :: GOSUB 6
000 :: X=5 :: RESTORE 5600 :
: GOSUB 7000
5020 RESTORE 5410 :: GOSUB 6
000 :: X=7 :: RESTORE 5610 :
: GOSUB 7000
5030 RESTORE 5420 :: GOSUB 6
000 :: X=5 :: RESTORE 5620 :
: GOSUB 7000
5040 RESTORE 5430 :: GOSUB 6
000 :: X=5 :: RESTORE 5630 :
: GOSUB 7000
5050 RESTORE 5440 :: GOSUB 6
000 :: X=5 :: RESTORE 5640 :
: GOSUB 7000
5060 RESTORE 5450 :: GOSUB 6
000 :: X=5 :: RESTORE 5650 :
: GOSUB 7000
5070 RESTORE 5460 :: GOSUB 6
000 :: X=8 :: RESTORE 5660 :
: GOSUB 7000
5080 RESTORE 5470 :: GOSUB 6
000 :: X=7 :: RESTORE 5670 :
: GOSUB 7000
5090 RESTORE 5480 :: GOSUB 6
000 :: X=6 :: RESTORE 5680 :
: GOSUB 7000
5100 RESTORE 5490 :: GOSUB 6
000 :: X=5 :: RESTORE 5690 :
: GOSUB 7000
5110 RESTORE 5500 :: GOSUB 6
000 :: X=7 :: RESTORE 5700 :
: GOSUB 7000
5120 RESTORE 5510 :: GOSUB 6
000 :: X=8 :: RESTORE 5710 :
: GOSUB 7000 :: CALL CLEAR
5130 RESTORE 5480 :: GOSUB 6
000 :: X=6 :: RESTORE 5680 :
: GOSUB 7000
5140 RESTORE 5490 :: GOSUB 6
000 :: X=5 :: RESTORE 5690 :
: GOSUB 7000
5150 RESTORE 5500 :: GOSUB 6
000 :: X=7 :: RESTORE 5700 :
: GOSUB 7000
5160 RESTORE 5510 :: GOSUB 6
000 :: X=8 :: RESTORE 5720 :
: GOSUB 7000 :: RETURN
5400 DATA DASH-ING THRU THE
SNOW,0
5410 DATA IN A ONE HORSE O-P
EN SLEIGH,2
5420 DATA O(ER THE FIELDS WE
 GO,2
```

```
5430 DATA LAUGH-ING ALL THE
WAY;,2
5440 DATA BELLS ON BOB-TAIL
RING,2
5450 DATA MAK-ING SPIR-ITS B
RIGHT,2
5460 DATA WHAT FUN IT IS TO
RIDE AND   SING,2
5470 DATA A SLEIGH-ING SONG
TO-NIGHT,3
5480 DATA JIN-GLE BELLS JIN-
GLE BELLS,2
5490 DATA JIN-GLE ALL THE WA
Y,2
5500 DATA OH WHAT FUN IT IS
TO RIDE,2
5510 DATA IN A ONE HORSE O-P
EN SLEIGH,2
5600 DATA 1,196,247,1,330,39
2,1,294,349,1,262,330,3,196,
247
5610 DATA .5,196,247,.5,196,
247,1,196,247,1,330,392,1,29
4,349,1,262,330,4,220,262
5620 DATA 1,220,262,1,349,44
0,1,330,392,1,294,349,4,247,
294
5630 DATA 1,392,494,1,392,49
4,1,349,440,1,294,349,4,330,
392
5640 DATA 1,196,247,1,330,39
2,1,294,349,1,262,330,4,196,
247
5650 DATA 1,196,247,1,330,39
2,1,294,349,1,262,330,3,220,
262
5660 DATA 1,220,262,1,220,26
2,1,349,440,1,330,392,1,294,
349,1,392,494,1,392,494,2,39
2,494
5670 DATA 1,392,494,1,440,32
3,1,392,494,1,349,440,1,294,
349,3,262,330,1,20000,20000
5680 DATA 1,330,392,1,330,39
2,2,330,392,1,330,392,1,330,
392,3,330,392
5690 DATA 1,330,392,1,392,49
4,1,262,330,1,294,349,4,330,
392
5700 DATA 1,349,440,1,349,44
0,1,349,440,1,349,440,1,349,
440,1,330,392,2,330,392
5710 DATA .5,330,392,.5,330,
392,1,330,392,1,294,349,1,29
4,349,1,330,392,2,294,349,2,
392,494
5720 DATA .5,330,392,.5,330,
392,1,392,494,1,392,494,1,34
9,440,1,294,349,3,262,330,1,
20000,20000
6000 READ L$,L :: Y=Y+L :: D
ISPLAY AT(1+Y,1):L$ :: RETUR
N
7000 FOR BT=1 TO X :: READ D
,T1,T2 :: CALL SOUND(M*D,T1,
3,T2,5,T1*2,9):: NEXT BT ::
RETURN
```

(PART 2 NEXT ISSUE)

STRUGGLING FORTH WITH RICHARD TERRY HV99

This week we will mainly look at string handling words - again home grown. To demonstrate their usage I will list several screens to make up a small program, study of which will also introduce the use of Menu's in Forth.

First of all the basic string words to accept strings, move them between addresses, add them (like concatenating in basic), compare identity, and find segments. If after analysing them you have a shorter version then please let us know.

BASIC STRING WORDS

REFER TO SOURCE CODE SCR#7

```
SCR #7
 0 ( STRING:Basic Word definitions 18Jly85)      ( STACK EFFECTS)
 1 : GET$  TIB @ SWAP EXPECT 0 IN ! 13 WORD      ( Adr to put new )
 2         HERE OVER OVER C@ DUP ROT C! 1+ !     ( string,count   )
 3         DO 1+ OVER OVER C@ SWAP I + C! LOOP   (      ----      )
 4         DROP DROP ;
 5 : MOVE$ SWAP COUNT 1+ SWAP 1- ROT ROT CMOVE ; ( Adr1 Adr2--- )
 6 : ADD$  DUP >R SWAP COUNT ROT COUNT DUP ROT
 7         + SWAP ROT DUP ROT + R> C! CMOVE ;    ( Adr1 Adr2-- )
 8 : SAME$ COUNT ROT COUNT ROT MAX 1 SWAP 0      ( Adr1 Adr2--flag )
 9         DO >R DUP C@ ROT DUP C@ ROT =         ( Where these adr )
10           R> MIN SWAP 1+ ROT 1+ ROT           ( contain strings )
11         LOOP >R DROP DROP R> ;                ( with counts     )
12 : SEG$  ROT DUP >R OVER SWAP C! SWAP ROT + R> ROT 0
13         DO OVER OVER SWAP C@ SWAP 1+ C! 1+ SWAP 1+ SWAP
14         LOOP DROP DROP ; ( From adr, adr to put, start,num--)
15 : GET$1 OVER OVER SWAP C! SWAP 1+ SWAP EXPECT ; ( adr,count---)
```

I will dissect one of these definitions to show how it works and give a brief description of the others.

All the above string words work on a dimensioned string - ie the first byte of the address containing the string has the character count Eg:

```
Address :BUF1
contents: 5 P E T E R
byte no : 1 2 3 4 5 6
```

We do not have to save it thus, but it is useful because forth words tend to presume a preceding character count. Eg to print the contents of the address above we would type:
TEMP COUNT TYPE to which forth would reply: PETER ok.

GET$1

This is the simplest version to place a packed string at an address, however it is not very adaptable as we must stipulate the character count ourselves.
Eg: TEMP 6 GET$1
If the user only types in a 3 byte word such as CAT the address will still contain 6CAT

GET$

This is a much smarter version (courtesy of El presidente himself!) It accepts the input into the TIB (terminal input buffer) and then uses WORD to check for a delimiter (In this case the ASCII 13 for a enter- this enables multiple words separated by spaces to be accepted, if you only desire a single input substitute an ASCII 32 or space character) does its own individual count for every time it is used and deposits the result in the buffer of your choice:

Eg TEMP 6 GET$ will accept up to 6 characters but will not panic if you only put in 3, the resulting count will be 3 not 6
Eg LOAD Scr# 7, type Ø VARIABLE TEMP 1Ø ALLOT TEMP 6 GET$, then type a 3 letter word eg CAT and press ENTER. To examine the contents of TEMP type TEMP C .
Even though you told it to accept up to 6 it saves the correct character count of three.

MOVE$

Simply uses CMOVE to move a dimensioned string. It expects:
From Address,To address on the stack

SEG$

This works the same as in Basic. It expects on the stack the address of the string you want to find a segment of, an address you allocate to store the segment, the spot along the string you want to start from, the character count to continue for.

TABLE 1

WORD :SAME$ Eg:Assume Adr1 has 3CAT Adr2 has 3DOG

```
DEFINITION: COUNT ROT COUNT ROT MAX 1 SWAP 0
-----------       DO    >R DUP C@ ROT DUP C@ ROT =
                        R> MIN SWAP 1+ ROT 1+ ROT
                  LOOP >R DROP DROP R> ;
```

WORD	PARAMETER STACK BEFORE	PARAMETER STACK AFTER	RTN STACK BEF	RTN STACK AFT
	Least accessible->most accessible			
COUNT	Adr1,Adr2	Adr1,Adr2+1,cnt2	---	---
ROT	Adr1,Adr2+1,cnt2	Adr2+1,cnt2,Adr1	---	---
COUNT	Adr2+1,cnt2,Adr1	Adr2+1,cnt2,Adr1+1,cnt1	---	---
ROT	Adr2+1,cnt2,Adr1+1,cnt1	Adr2+1,Adr1+1,cnt1,cnt2	---	---
MAX	Adr2+1,Adr1+1,cnt1,cnt2	Adr2+1,Adr1+1,maxcount	---	---
1	Adr2+1,Adr1+1,maxcount	Adr2+1,Adr1+1,maxcount,1	---	---
SWAP	Adr2+1,Adr1+1,maxcount,1	Adr2+1,Adr1+1,1,maxcount	---	---
0	Adr2+1,Adr1+1,1,maxcount	Adr2+1,Adr1+1,1,maxcount,0	---	---
DO	Adr2+1,Adr1+1,1,maxcount,0	Adr2+1,Adr1+1,1	---	---
>R	Adr2+1,Adr1+1,1	Adr2+1,Adr1+1	---	1
DUP	Adr2+1,Adr1+1	Adr2+1,Adr1+1,Adr1+1	---	1
C@	Adr2+1,Adr1+1,Adr1+1	Adr2+1,Adr1+1,"C-67"	---	1
ROT	Adr2+1,Adr1+1,"C-67	Adr1+1,"C-67",Adr2+1	---	1
DUP	Adr1+1,"C-67",Adr2+1	Adr1+1,"C-67",Adr2+1,Adr2+1	---	1
C@	Adr1+1,"C-67",Adr2+1,Adr2+1	Adr1+1,"C-67",Adr2+1,"D-68"	---	1
ROT	Adr1+1,"C-67",Adr2+1,"D-68"	Adr1+1,Adr2+1,"D-68","C-67"	---	1
=	Adr1+1,Adr2+1,"D-68","C-67"	Adr1+1,Adr2+1,0(unequal)	---	1
R>	Adr1+1,Adr2+1,0	Adr1+1,Adr2+1,0,1	---	---
MIN	Adr1+1,Adr2+1,0,1	Adr1+1,Adr2+1,0	---	---
SWAP	Adr1+1,Adr2+1,0	Adr1+1,0,Adr2+1	---	---
1+	Adr1+1,0,Adr2+1	Adr1+1,0,Adr2+2	---	---
ROT	Adr1+1,0,Adr2+2	0,Adr2+2,Adr1+1	---	---
1+	0,Adr2+2,Adr1+1	0,Adr2+2,Adr1+2	---	---
ROT	0,Adr2+2,Adr1+2	Adr2+2,Adr1+2,0	---	---
LOOP	Adr2+2,Adr1+2,0	Adr2+2,Adr1+2,0	---	---

Time passes....(quickly) whilst the loop loops until suddenly we exit.... only to find our addresses are terminal and that our trusty Ti994a spews FORTH the answer that a CAT is not the same as a DOG!

>R	Adr2+4,Adr1+4,0	Adr2+4,Adr1+4	---	0
DROP	Adr2+4,Adr1+4	Adr2+4	0	0
DROP	Adr2+4,		0	---
R>		0 (resultant false flag)		

----------END OF DEFINITION----------

SAME$

I will analyse this one in detail to show how it compares strings. if Adr1 has say DOG
Adr2 has say DOG
then it will leave a True flag
ie 1 since they are identical. It leaves a false flag ie Ø if the words being compared are different. To actually see what happens to the flags with various words, LOAD screen 14 to see it in action.

We will examine in complete detail the stack during the execution of SAME$. Refer to Table 1 for a blow by blow description of the stack, inconjunction with the following description.

SAME$ works by initially getting the Byte counts of the two words, taking the largest for the DO LOOP indices. Giving the words to be compared the benefit of the doubt it assumes equality by starting with a true

flag(1). Each byte of each word is compared giving a resultant flag:1 if letters are the same or Ø if they are different. This flag is compared to the one we hide on the return stack using MIN and the lowest result kept. Hence as soon as the letters are unequal the initial true flag(1) will change to a false flag(Ø) for ever after. (One could change the definition to add a comparision to Ø and the option of leaving the loop upon this happening)

COUNT ROT COUNT ROT
-obtains the character counts of both addresses leaving them on top of the stack

MAX
- leaves the bigger of the two for the loop index

1
- our true flag to start with

SWAP Ø
-leaves the loop indices on top of the stack for use by DO LOOP

>R
-saves to return stack for later comparision our true flag

DUP C
-duplicates start address, fetches the ASCII code of the first letter, in this case a C-ASCII 67

ROT DUP C
-Rotates the second address to the top and does the same thing, in this case resulting in a D-68

ROT =
-brings the two ASCII codes to the top of the stack and checks if they are the same leaving a False (Ø) flag

R> MIN
-brings our initial true flag back from return stack,and leaves the lesser flag (Ø)

SWAP 1+ ROT 1+ ROT
-Increments our two starting addresses leaving the flag for comparison on top of the stack so that on the next loop it will be immediately saved to the return stack as above.

The loop will then repeat three times and the resultant end addresses will be Adr+4.

>R
-once the loop is finished the resultant flag is again temporarily saved on the return stack whilst the two terminal addresses we no longer need are dropped leaving a clean stack with only the flag on top, for use by the rest of the program. Now some further comments on the rest of the screens which make up the string word example.

REFER TO SCR#8

```
SCR 08
 0 ( STRING EXAMPLE -Messages IBJly85)
 1
 2 : CONTINUE BEGIN ?KEY 32 = UNTIL ;
 3
 4 : MSG1 5 12 GOTOXY ." Moving Strings" 5 14 GOTOXY
 5       ." From Buf1 to Buf2" ;
 6 : MSG2 5 12 GOTOXY ." Adding Strings" 5 14 GOTOXY
 7       ." From Buf1 to Buf2" ;
 8 : MSG3 0 23 GOTOXY ." Press any key" CONTINUE ;
 9 : MSG4 5 12 GOTOXY ." Comparing Strings" 5 14 GOTOXY
10       ." In Buf1 to Buf2" ;
11
12
13
14
15
```

CONTINUE

BEGIN(s) a keyboard scan (?KEY) UNTIL the ascii 32 (space bar) has been pressed ie it holds up the program until user decides to CONTINUE. This word is better saved into your core dictionary along with the string words.

I have named my screen prompts as messages (MSG), keeping them together on a single screen for neatness and ease of reference. Another way to put up screen prompts is to us the word MESSAGE which will read a line on any screen relative to screen 4, line Ø. To make it easy for you:

LINE=(Scr#-4)+line(ofscreen containing the message.

REFER TO SCR#9

Scr#9 allocates space for our bufers BUF1, BUF2
ENTER$ ie allow string entry GETBOTH puts up the PROMPT, blanks the buffers with BLANKBUFS and uses GET$ to transfer packed strings to BUF1,BUF2. MSG3 asks to press a key (inaccurately as it only likes a

```
SCR 09
 0 ( STRING EXAMPLE -Accepting   18Jly85)
 1 0 VARIABLE BUF1 20 ALLOT   0 VARIABLE BUF2 20 ALLOT
 2 : BLOT  0 12 160 32 HCHAR  0 23 40 32 HCHAR  5 20 15 32 HCHAR
 3         5 21  15 32 HCHAR 25 20 15 32 HCHAR 25 21 15 32 HCHAR ;
 4 : PROMPT 5 12  GOTOXY ." Enter Strings"
 5          5 14  GOTOXY ." First  string:"
 6          5 15  GOTOXY ." Second String:" ;
 7 : BLANKBUFS BUF1 22 BLANKS BUF2 22 BLANKS ;
 8 : GETBOTH BLANKBUFS PROMPT  19 14 20 32 HCHAR 19 15 20 32 HCHAR
 9                             19 14 GOTOXY BUF1 10  GET$
10                             19 15 GOTOXY BUF2 10  GET$ ;
11 : BUFAFTER 25 20 GOTOXY BUF1 COUNT TYPE
12            25 21 GOTOXY BUF2 COUNT TYPE ;
13 : ENTER$ GETBOTH BUFAFTER MSG3 BLOT  ;
14
15
```

space bar-you can easily fix this yourself) and then BLOTS out the prompts etc.

REFER TO SCR#10,SCR#11

```
SCR 10
 0 ( STRINGS EXAMPLE -Move & Add 18Jly85)
 1 : BUFBEFORE 5 20 GOTOXY BUF1 COUNT TYPE
 2             5 21 GOTOXY BUF2 COUNT TYPE ;
 3 : DELAY 10000 0 DO NOP LOOP ;
 4 : ?SAME$  MSG4 BUFBEFORE MSG3 CONTINUE BUF1 BUF2 SAME$ BUFAFTER
 5           5 15 GOTOXY 1 = IF ." STRINGS IDENTICAL" ELSE
 6           ." STRINGS DIFFERENT" THEN MSG3 DELAY CONTINUE BLOT ;
 7 : DOMOVE MSG1 BUFBEFORE MSG3 BUF1 BUF2 MOVE$ BUFAFTER
 8          DELAY   MSG3 BLOT ;
 9 : ADDTHEN MSG2 BUFBEFORE MSG3 BUF1 BUF2 ADD$ BUFAFTER
10          DELAY MSG3 BLOT ;
11 : BUFSCREEN  0 18 GOTOXY ." Contents Before"
12             20 18 GOTOXY ." Contents After"
13              0 20 GOTOXY ." Buf1:" 20 20 GOTOXY ." Buf1:"
14              0 21 GOTOXY ." Buf2:" 20 21 GOTOXY ." Buf2:" ;
15
```

```
SCR 11
 0 ( STRING EXAMPLE -Subsegments 18Jly85)
 1   VARIABLE FROM 2 ALLOT 0 VARIABLE TO 2 ALLOT
 2
 3 : SEGHEADING 5 12 GOTOXY ." String Segments"
 4              5 13 GOTOXY ." Enter String:"  BUF1 15 GET$
 5             5 14 GOTOXY ." Segment from:  For:   Characters " ;
 6 : ?FROM 19 14 GOTOXY KEY DUP EMIT 48 - FROM ! ;
 7 : ?TO   25 14 GOTOXY KEY DUP EMIT 48 - TO   ! ;
 8
 9
10
11
12 : SEGMENT SEGHEADING ?FROM ?TO  BUF1 BUF2 FROM @ TO @ SEG$
13          5 15 GOTOXY ." Segment is:" BUF2 COUNT TYPE CONTINUE
14          BLOT ;
15
```

Scr#10,Scr#11 are simply more words to deal with making the screen look nice and orderly to show adding/moving examples and hopefully are self explanatory.

REFER TO SCR#12

This screen is a useful format for a basic menu driven program. It prints a aesthetically pleasing menu and uses CHOICE to obtain you option. Each time your choice is examined it is DUPlicated so it will not be lost forever!

```
SCR 12
 0 ( STRING EXAMPLE -Main menu 18Jly85)
 1 : OPTIONS CLS  5 0  GOTOXY ." STRING HANDLING WORDS"
 2                5 2  GOTOXY ." Select Option:"
 3                5 4  GOTOXY ." 1.Enter Strings"
 4                5 5  GOTOXY ." 2.Move from buf1 to buf2"
 5                5 6  GOTOXY ." 3.Compare Strings"
 6                5 7  GOTOXY ." 4.Concatenate-join Strings"
 7          5 8  GOTOXY ." 5.String Segment" 5 9 GOTOXY ." 6.End" ;
 8 : CHOICE  19 2 GOTOXY KEY DUP EMIT 48 -
 9   DUP 1 = IF ENTER$ ELSE DUP 2 = IF DOMOVE    ELSE
10           DUP 3 = IF ?SAME$    ELSE
11           DUP 4 = IF ADDTHEN   ELSE
12           DUP 5 = IF SEGMENT   ELSE
13           DUP 6 = IF PAGE QUIT ELSE
14           THEN THEN THEN THEN THEN THEN DROP MYSELF ;
15 : RUN OPTIONS BUFSCREEN CHOICE ;
```

REFER TO SCR#13

This LOAD(s) sequentially our program . Type RUN to start the program.

```
SCR 13
 0 ( STRING EXAMPLE -Load screen)
 1
 2
 3 PAGE ." LOADING STRING EXAMPLE-PLEASE WAIT"
 4
 5  7 LOAD  8 LOAD 9 LOAD 10 LOAD
 6 11 LOAD 12 LOAD
 7
 8
 9
10
11
12
13
14
15
```

Note how simple our final definition is consisting of three words:
OPTIONS / BUFSCREEN put up the available options and the buffer contents before use
CHOICE directs the program to the desired option and once run back to the menu.
To see the whole thing work FORGET back to the start of your core words and type 13 LOAD and then RUN.As these are direct printouts from my Forth screens it should be fairly accurate as it worked for us at our FIG meeting, but if you encounter any

```
SCR #14
 0 ( STRINGS ?IDENTICAL 1531y85)              ( STACK EFFECTS)
 1 7 LOAD 8 LOAD 9 LOAD
 2 : SAME$1 COUNT ROT COUNT ROT
 3         MAX 1 SWAP 0 DO                    ( Adr1 Adr2--flag)
 4         DO )R DUP C@ DUP EMIT 6 SPACES
 5            ROT DUP C@ DUP EMIT
 6            ROT - 6 SPACES  DUP .
 7            5 SPACES
 8           R> MIN DUP . CR
 9           SWAP 1+ ROT 1+ ROT          ( contain strings )
10         LOOP )R DROP DROP R> ;              ( with counts )
11 : HEADING 0 0 GOTOXY  ." WORD1 WORD2  FLAG  MIN" CR ;
12 : RESULT CR  1 = IF ." STRINGS IDENTICAL" ELSE
13                  ." DIFFERENT STRINGS" THEN ;
14
15 : $TEST  CLS GETBOTH  HEADING BUF1 BUF2 SAME$ RESULT ;
```

problems ring me on either 436511/22450.

Next month with any luck a small program to save an array of names , print them to a BASIC file and retrieve them, along with a BASIC program to do the same.

PS:A WORD OF WARNING. -There is not much error checking in this program.If you attempt to keep moving strings into the same buffer you will write over the next dictionary definitions and LOCK UP THE COMPUTER. If you enter a null string this will not be detected and cause unpredictable results.

Richard Terry 30/10/85.

****
* BUGS-BUGS-BUGS-BUGS-BUGS-BUGS *
****

A bug in MINI FORMATTER has raised its ugly head. It seems that with some systems if the printer is not turned on the programme crashes. And I am indebted to Mr. Kleinschafer for this information and the type of changes to fix it. To fix it just alter line 460 to the one below and add line 465.
Brian R.

```
460 DISPLAY AT(3,18)BEEP:"PI
O" :: ACCEPT AT(8,18)SIZE(-3
):P$ :: IF P$="" THEN 460
465 CALL SCREEN(14):: DISPLA
Y AT(22,1)BEEP:"Is your prin
ter turned on!":"" :: CALL K
C :: CALL SCREEN(5):: OPEN #
1:P$ :: IF K=49 THEN SUBEXIT
```

ENTOMOLOGY CORNER. FROM FUNNELWEB FARM

BY

TONY MCGOVERN AV99

It looks like the time has come once again actually to use Funlwriter and desist from assembly programming before Steve has to wrap up the magazine for this issue. I have been so absorbed with Version 2.2 of FUNLWRITER that most other things have been put aside. I must confess that I haven't been writing much at all in Extended Basic, using it mainly as an auto-loading lead in to assembly code. It will all be good stuff for Tutorial articles in the future. That's how the present almost completed series got its start (yes I know it isn't quite done with yet), from a earlier period of intense concentration. Programming is like that, you don't master it without a lot of effort. Oh well, just so long as it remains an absorbing challenge. Extracting the maximum from the machine is the only, certainly the best, home computer game there is.

The warning flags remain set for Disk Manager 1000. I have written to the author but no reply has yet been received. Don't know whether this means that he is buried deep in bug-hunting, or has gone into hiding in deep shame, or has just abandoned the TI-99/4a for greener pastures. That's something that is a lot more difficult to do in Australia with the depressed currency added to traditional importing duties and rip-offs, and non-competitive

wholesale and retail markups. I just don't see anything available here now that looks better value than our present TI system. Can you imagine descending to 6502 family 8-bit coding, a la Apple or Commodore, after getting used to the TMS-9900. I know as of today that Funlwriter is extant in Ottawa from an air-mail letter that took a month to get here. Perhaps a reply from the author is buried even further down in that great push-down stack of the mails in Sydney.

The name 'FUNLWRITER' is now an inadequate indication of what the program can do, but I can't think of an alternative. Too busy working on it! You can now go from the Formatter back to the Editor without going through the title screen or losing your last file name. As promised last issue, Funlwriter now completely supports operation of the TI 99/4 assembler as provided on the E/A disk, with an improved user interface. It is also possible to go back from the assembler directly to the source code editor,(TI-Writer customized for the purpose), and not lose the source file name. I have not tried to emulate the E/A screens, and Funlwriter does things in its own style. There were some very perplexing difficulties encountered in making the assembler operation fully re-entrant. I eventually traced the problem to a bug in TI's assembler code, patchable on the fly by Funlwriter. This bug is not stirred up in the usual operation of the assembler, else TI would have noticed it and fixed it. But it is still a bit of fun to find something that TI's programmers blew it on. I suspect it might be possible to stir this bug up to crash the assembler with a legal input, but I haven't found the time to try it yet.

One thing that can be said for TI, despite all their sins of commission and omission on the 99/4a, is that they did do a thoroughly professional job of getting their software free of harmful bugs before release. Maybe that's how come it was always so slow to appear. This was brought home to me on reading the Oct/85 issue of Your Computer in the page on BBC machines describing the state of the manufacturer supplied word processor, still bug ridden in revised versions. The problems of dealing with the supplier sounded vaguely familiar though.

So what else is new in Funlwriter. Well it now has an E/A LOAD RUN option which will load E/A object files with full utility support including GPLLNK. No more having to guess the program name to RUN either. Funlwriter puts the DEF table on screen with cursor driven selection. That's something TI should have done originally, just as they should and could easily have provided the other enhancements which have had to wait for Funlwriter. A GPLLNK is now also provided as an additional XB utility as well as DSRLNK. This work has also brought to light a bug in the E/A manual. Not that it doesn't have lots of others, but it was one I was taking as gospel without detailed confirmation. This is the statement in Section 24.11 that auto starting object files (with entry point defined on the END directive) start out in the GPL workspace. This is not true, and all object files loaded with the E/A module, whether from E/A or Basic, start out in the USRWSP at >20BA. The only exception to this comes if the object file is loaded directly from assembly code with BLWP @LOADER, not using the XML routine invoked by the module. In contrast TI-Writer and XB hand over to assembly routines in the GPL workspace. For the record, the Miller's Graphics Explorer which we had briefly for review before last issue is also incorrect on this point. I might be maligning them, as I no longer have the book or program to do a last check on, but I distinctly recall the assertion that the USRWSP is not loaded. The other comments there on the differing departure points from the module for auto-start and normal entry are correct. It may also be that TI issued E/A in two or more distinct versions without bothering to tell users of changes. There are reportedly many more console models then is apparent on the surface, so why not modules too ? The E/A manual also states very positively and just as incorrectly that auto starting object files also do this when CALL LOADed from XB. Not on this module they don't. This does not affect Funlwriter because it uses its own assembly code to manage the handover, not relying on inaccessible GPL routines in the module.

I had thought that the status of Funlwriter as a replacement for the TI-Writer module would define implicitly the system level needed to run it, but apparently that has not been adequate. For the record now, Funlwriter requires Extended Basic in an expanded system with 32K and disk, and also a printer interface (eg RS232) if print-out is wanted. Corcomp systems work as well as they usually do. The assumptions made about the system are the same as those made by TI-FORTH, and it has been a consistent design principle to avoid detail beyond that, such as funny disks incompatible with Disk Manager or direct accessing of the disk controller chip. In this connection I recall reading that there was a third party disk system in the US of A that did not support TI-FORTH but it is most unlikely to be found in Australia. The performance level obtained without funny business (otherwise known as using undocumented accessing to enhance performance) is hardly inferior to the original. Needless to say, I regard disk protection as an abomination when applied to any program that pretends to be useful.

Will has been beavering away as light and too frequent relief from 10th year exam study and has created an item that will be of interest to group members who are installing the internal 32K memory expansion in their consoles. This is an Extended Basic program which itself loads and runs from cassette and then loads and runs an assembly program file from cassette. The last one he demonstrated to me was the TI DEBUG suitably prepared as a program file, loaded from cassette. Of course E/A can RUN PROGRAM FILE from cassette, but only people who already own a full system are likely to have E/A, while anyone with enough interest and enthusiasm to install the 32K no doubt already has XB. His latest twist has been to add another option to DEBUG which allows saving to disk in E/A SAVE format of a designated memory block.

That's about all the gossip and progress report we have time for now. Steve is planning a special Xmas issue so we will have to get something together that shares the experiences and lessons of the last few months programming work.

-:: WV99 LIBRARY NEWS ::-

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

Well we are well on our way into Adventureland and are getting near solving our first Scott Adams Adv.
Sorry we have only had 2 nights but as those members who have been there will have seen soldering irons in use to get the 32K boards ready for the lucky owners who are expanding their machines to greater use and we have some good news on programs for your advantage as WILL McGOVERN will demo at the meeting!!!!.
We are trying to get together all the hints and tricks on solving some of the adventures so if you come across any please pass them on to us to compile.

Disk news

We received some good disks to add to our library some of these are on this months cassette they are also available as disks lots on if you would like these let me know.

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 90 disks.
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. Cassette are still a good buy at the WV99'er monthly meet for $ 7.00.

That's it for now,
Happy programing,
Al Lawrence.

STOP PRESS. STOP PRESS. STOP PRESS. STOP PRESS

Brian Rutherford has a horse racing program which he would like some of the more serious punters to test out. If you are familiar with horse racing and would like to help Brian debug the program give him a call on 498184 or see him at the meeting.

Alan Lawrence is organising a Christmas picnic/barby for December somewhere on the shores of beautiful Lake Macquarie. Transportation in the form of a double decker bus will be provided; which is sure to be a big hit with the young children. If you would like to come along and get to meet the families of your fellow TI99/4A fanatics and share a few pleasant hours away from the console let Al know by phone on 468509 or see him at the meeting.

WANTED: Tony Mc.Govern is still after a stand alone disk controller unit. If you have one or know of someone who does give Tony a call on 523162

FOR SALE: Al Lawrence has his NAVARONE DISK FIXER MODULE for sale. Ph. 468509

FOR SALE: One stand alone MPI disk drive. This drive is only a few months old and in as new condition. It has a built in power supply so it is just a matter of connecting up the cable that came with your PE BOX and away you go. If you are interested you can have it for $150.00 but if you twist my arm you may get it for less. Call Steve on 487076

FOR SALE: Jim Threadgate has a complete TI system for sale. It includes a console, PEB, RS232, 32K., disk drive and disk controller card plus a large assortment of modules and software. Call Jim on 335970

FOR SALE: Navarone Widgit. Half price, $30.00. call Steve on 487076

WANTED: Modules for the module library. See Bob Maclure

STOP PRESS. STOP PRESS. STOP PRESS. STOP PRESS