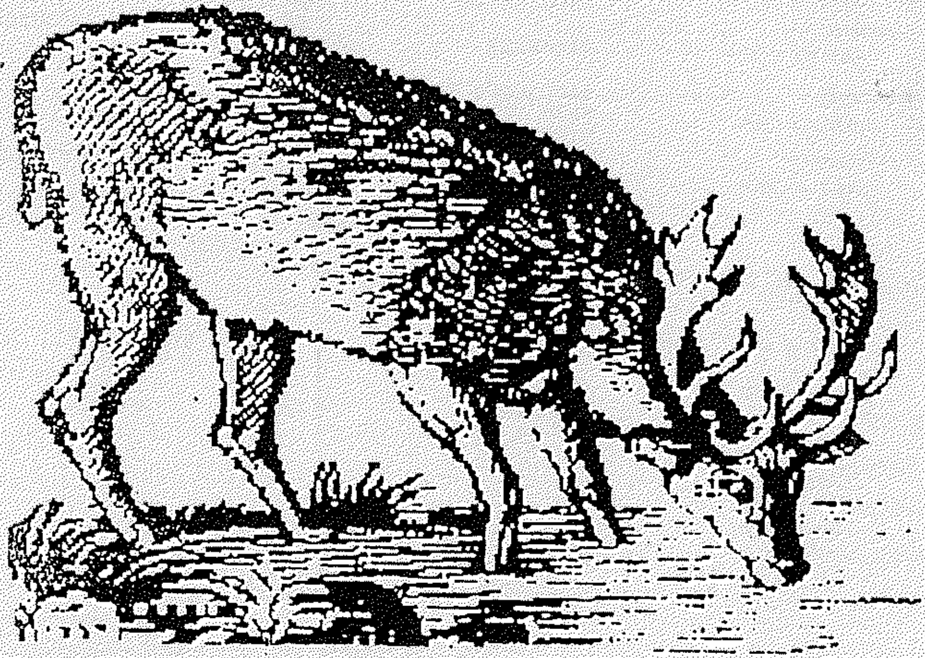
TISHUG

NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 13, Number 7 August, 1994

Print Post Approved - Publication No. PP244099/00016



Sydney, New South Wales, Australia $3

TIsHUG (Australia) Ltd.
A.C.N. 003 374 383

TIsHUG News Digest

All correspondence to:
C/o 3 Storey St.
Ryde 2112 Australia

The Board

Co-ordinator	
Dick Warburton	(02) 918 8132
Secretary	
Robert Relyea	(046) 57 1253
Treasurer	
Cyril Bohlsen	(02) 639 5847
Directors	
Percy Harrison	(02) 808 3181
Thomas Marshall	(02) 871 7535

Sub-committees

News Digest Editor	
Loren West	(047) 21 3739
BBS Sysop	
Ross Mudie	(02) 456 2122
BBS telephone number	(02) 456 4606
Merchandising	
Percy Harrison	(02) 808 3181
Software Library	
Larry Saunders	(02) 644 7377
Technical Co-ordinator	
Geoff Trott	(042) 29 6629

Regional Group Contacts

Central Coast	
Russell Welham	(043) 92 4000
Glebe	
Mike Slattery	(02) 692 8162
Hunter Valley	
Geoff Phillips	(049) 42 8176
Illawarra	
Geoff Trott	(042) 29 6629
Liverpool	
Larry Saunders	(02) 644 7377
Sutherland	
Peter Young	(02) 528 8775

Membership and Subscriptions

Annual Family Dues	$35.00
Associate membership	$10.00
Overseas Airmail Dues	A$65.00
Overseas Surface Dues	A$50.00

TIsHUG Sydney Meeting

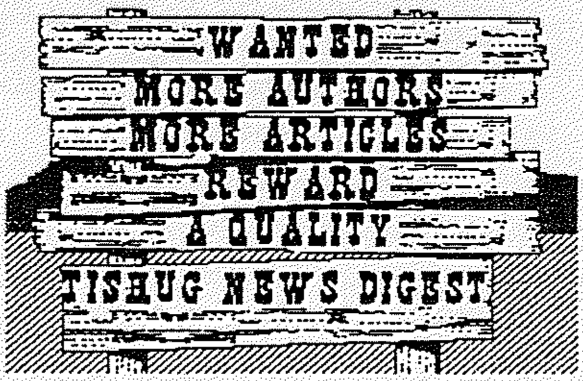
The August Meeting will start at 2.0 pm on the 6th August 1994 at Meadowbank Primary School, Thistle Street, Meadowbank.

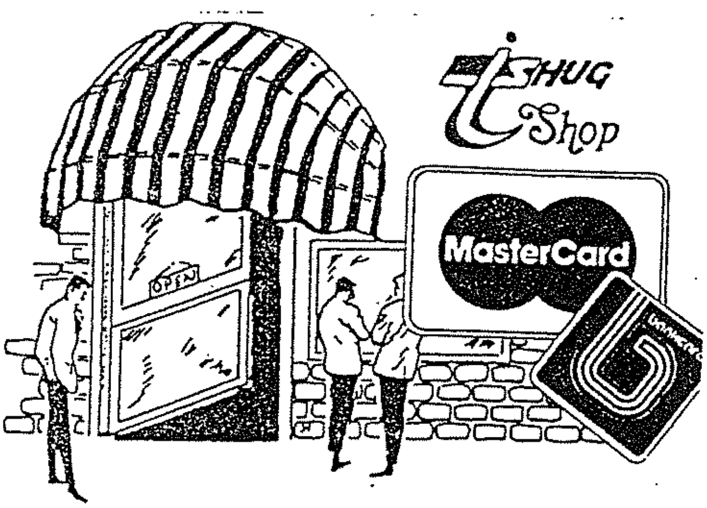
Printed by
Kwik Kopy West Ryde

TIsHUG News Digest ISSN 0819-1984

INDEX

IBM INDEX





TIsHUG SHOP.

with Percy Harrison.

Unfortunately I was not able to attend the July meeting as my wife and I had promised to take the two grandchildren away for the school holidays. The weather was rather unkind to us as it rained nearly every day but the holiday cottage was very comfortable with great views of the ocean and mountains and the table tennis facilities certainly helped to keep the children occupied.

The shop was left in the capable hands of Cyril Bohlsen who did an excellent job selling software for both the TI and IBM compatible machines, his only complaint being that it prevented him from getting to see, and participate in, the events that were going on elsewhere at the meeting.

Progress on the TIM/SOB cards has been very slow as Dick Warburton is still trying to source components at the very best price in order to keep our losses down to a minimum. About half the components have already been purchased and hopefully the balance will be in hand before our next meeting. A small group will then be established to assemble the first 20 cards. For those members who have not already paid for the TIM/SOB cards please note that at this stage I do not require a deposit but I do need a firm commitment that you want, and will take, the cards when they become available. I will keep you informed on the progress of these units through this column each month.

Once again, on behalf of the directors, I would like to apologise for the poor print quality of the July TND. Our usual printer was unable to do the printing for us and subcontracted it to another printer. The poor quality was not noticed until we had completed inserting the magazines into the envelopes for mailing and by that time it was too late to do anything about the problem. We are currently trying to source an alternate printer who would be more reliable and who will guarantee a higher quality reproduction at the same or lower cost to us.

Our apologies also for an error in Larry Saunders July Software File. The diskname P083 should have been U083 as it is a utilities disk not a Page Pro Pictures disk as the prefix would indicate. Any member who purchased this disk expecting it to be full of Page Pro Pictures may exchange it at the next meeting if they so desire.

We are having difficulty in finding enough suitable articles on IBM compatible machines and programs to ensure a fair proportion of our magazine is allocated to our "other members" so come on you IBM compatible experts let's have some articles from you that are suitable for our magazine. On this score I think Ashley Lynn is to be commended for his July article on "Windows". To my recollection it is the first IBM compatible article, by one of our members, to be published in our magazine.

Finally, I would like to thank one of our ex members, Lou Armadio, for his donation of several modules and books to the club shop. Lou contributed many technical articles to our magazine back in the days when he was an active member and freely gave his technical expertise and advice for the benefit of many of our members. I note that Lou has attended our monthly meetings twice in the last three months so perhaps with a little coaxing he can be enticed to rejoin the club and share some of the expertise he has on IBM compatibles with other members and so encourage the development of this side of our club. Could some of our Wollongong members please pass our thanks on to Lou.

PRICE LIST.

TI MODULES AVAILABLE

Adventureland Cassette + Book	$5
Beginners Basic Tutor Cass. + Book	$4
CART-WRITER Module Only	$20
DIAGNOSTICS Module Only	$20
Disk Manager 2 Module Only	$10
Hunt the Wumpus Module + Book	$8
Hustle Module + Book	$8
Mission Impossible Cassette + Book	$5
Munchman Module + Book	$8
Paint 'N Print Module Only	$25
Parsec Module + Book	$8
Teach Yourself Basic Cassette + Book	$4
Terminal Emulator II Module + Book	$20
TI Invaders Module + Book	$8
TI-Writer Word Processor Module + Book	$25
Tris Module Only	$25
TI Logo Module + Book	$8
Video Graphs Module + Book	$8
Video Chess Module + Book	$8

Bye for now.

END OF ARTICLE

TREASURER'S REPORT

by Cyril Bohlsen

Income for previous month	$ 369.00
Expenditure for previous month ..	$ 1335.84
Loss for previous month	$ 966.84

Membership accounted for $ 105.00 of Income.

Shop sales .... $ 264.00 of Income.

The expenditure was made up of the following

Printing & Postage of TND	$ 307.54
Purchase of 80 Column Card Parts .	$ 400.00
Purchase of B/J Printer Ink	$ 112.50
Purchase of Envelopes for TND	$ 446.00
BBS Running Expenses	$ 69.80

TISHUG Software August 1994

By Larry Saunders

```
--------------------------------------------
Diskname P085
Total Sectors 358 Free Sectors 9
Date AUG 1994 Files 14

Page Pro Pictures

     BUCKLEY      16 I 13  COFFEEPOT    10 I 13
     ENTERPRISE   36 I 13  EPLURIBUS    20 I 13
     F14          67 I 13  FLAGCANNON   14 I 13
     FLAGRAISER   14 I 13  FRANKLIN     28 I 13
     GRANT        19 I 13  HENDRIX      32 I 13
     HHOUSE       25 I 13  HORSESHOE    16 I 13
     JACKSON      26 I 13  JEFFDAVIS    26 I 13
--------------------------------------------
Diskname P086
Total Sectors 358 Free Sectors 1
Date AUG 1994 Files 17

Page Pro Pictures.  Note BLTARC is Archived.

     ALASKA       48 I 13  BARBECUE     19 I 13
     BATMAN       25 I 13  BLTARC       27 I128
     BRUSH         7 I 13  BUCKLEY      16 I 13
     CASPER       24 I 13  COUPLE       27 I 13
     COWBOYS      11 I 13  CUPID2       24 I 13
     DECOY        10 I 13  DODGERS       9 I 13
     EAGLE1       13 I 13  EAGLE2       67 I 13
     EAGLE3       18 I 13  FIFEDRUM2     5 I 13
     FLAG          7 I 13
--------------------------------------------
Diskname P087
Total Sectors 358 Free Sectors 6
Date AUG 1994 Files 14

Page Pro Pictures

     LANTERN       8 I 13  SOLDIER3     34 I 13
     STAMP1       22 I 13  STAMP2       41 I 13
     STAMP3       22 I 13  SWAN         47 I 13
     TROLLEY      28 I 13  USSEAL       32 I 13
     WASHINGTON   26 I 13  WHEELBARRW   24 I 13
     XGARFIELD1   15 I 13  XGARFIELD2   16 I 13
     XGARFIELD3   18 I 13  XGARFIELD4   19 I 13
--------------------------------------------
Diskname P088
Total Sectors 358 Free Sectors 1
Date AUG 1994 Files 20

Page Pro Pictures

     JEFFERSON    26 I 13  LADYFLAG     18 I 13
     LBELL         4 I 13  LINCOLN1     22 I 13
     LINCOLN2     27 I 13  LIONS         9 I 13
     MADEINUSA     5 I 13  MADEINUSA2    4 I 13
     MARTIAN      49 I 13  METS         10 I 13
     MRMAGOO      23 I 13  PICARD       44 I 13
     PINELLA      16 I 13  REDSOX        8 I 13
     RELEE1       34 I 13  RELEE2       28 I 13
     SAILOR        7 I 13  SHIELD        8 I 13
     SOLDIER       3 M 57
```

LEARN TO KNOW YOUR TI LESSON 18

with Percy Harrison

This lesson explains the Command Mode and the Run Mode of the computer. Perhaps it should have been dealt with earlier in the course but I purposely put it off until now in order to get you familiar with the other core commands in BASIC.

The only commands that we will use in this lesson are:

NEW, PRINT and RUN

Other names for these modes are:

Command Mode: direct mode immediate mode

Run Mode: deferred mode

In some computers, the command mode is called "the edit mode" but TI BASIC has a line edit mode called "the edit mode".

In the command mode you enter a line and the characters go into the input buffer. When ENTER is pressed, the computer looks to see if the line starts with a number. If so, it stores the line in the program space, making room at the right location so that the lines are numbered in order.

A line must start with a line number followed by a "statement" key word. Or it must start with no number and a "command" key word, otherwise, an error message will be printed on pressing ENTER.

LESSON 18 COMMAND AND RUN MODES

Enter: NEW

You are ready to begin the lesson.

EXECUTION AND RUNNING

We mean "execution" like the soldier executing a command "Left Face", not "execution by firing squad".

"Execute a program" means the same as "run a program".

PROGRAM EXECUTION

Enter and run this program:

```
10 PRINT "HI"
```

This is the usual way to make and run programs. You enter program lines.

Each line starts with a number, followed by a statement. The computer stores the line with the other lines in memory. Later you execute the program by entering the command "RUN".

COMMAND EXECUTION

Here is a short cut. Enter this (no line number in the front):

```
PRINT "HI"
```

This time PRINT is used as a command. The computer executes the command right away without waiting for you to enter RUN. It does not store the command in its memory so once you press ENTER the command is executed once only and is then lost.

ASLEEP OR AWAKE

People act one way if they are awake and another way if they are asleep. They have two "operating modes".

You can tell if they are asleep because they snore. (Well, not all people snore, but to explain how computers are like people, let's pretend that all sleeping people snore). The computer has two operating modes too. They are called the "command mode" and the "run mode".

THE COMMAND MODE

Enter: NEW

You see the ">" symbol and the flashing cursor square.

The ">" is called a "prompt" and says that the computer is in the "command mode" of TI BASIC.

The ">" is the "snoring" of the computer when it is in the command mode.

In the command mode the computer just waits for you to enter something and because it is very patient it will keep on waiting until it receives an input from you.

While the computer is in the command mode:

> You can enter programs by typing lines which start with numbers.
> You can use the computer like a pocket calculator. Big pocket hey!
> You can enter commands like LIST,CALL CLEAR, PRINT, etc.

THE CALCULATOR

You can do arithmetic in the Command Mode. Try this:

```
PRINT 3+7
```

Press ENTER and the computer prints the answer "10".

THE RUN MODE

Enter RUN to leave the command mode and go into the run mode.

> While the computer is in the run mode:
>
> The screen turns green.
>
> The program in memory runs.

When the program is finished, the computer automatically goes back to the command mode.

Assignment 18:

1. How can you tell if the computer is in the command mode?
2. How can you tell if the computer is in the RUN mode?
3. What mode does the computer enter when the program is done running?
4. How can you tell where the next character you type will appear on the screen?

ANSWERS TO LESSON 17

Assignment Question 17-1

```
5 REM -- FILM STAR
10 CALL CLEAR
15 LET C$="DEBORAH KERR"
20 LET D$="THE KING AND I"
25 PRINT TAB(9);C$
30 PRINT
31 PRINT
35 PRINT TAB(8);D$
40 FOR Z=1 TO 11
45 PRINT
50 NEXT Z
55 CALL SCREEN(2)
90 FOR HUE=3 TO 15
100 CALL COLOR(5,HUE+1,2)
101 CALL COLOR(6,HUE+1,2)
102 CALL COLOR(7,HUE+1,2)
110 FOR DELAY=1 TO 150
120 NEXT DELAY
130 NEXT HUE
140 GOTO 90
```

Assignment Question 17-2

```
5 REM -- FILM STAR
10 CALL CLEAR
15 LET C$="DEBORAH KERR"
20 LET D$="THE KING AND I"
25 PRINT TAB(9);C$
30 PRINT
31 PRINT
35 PRINT TAB(8);D$
40 FOR Z=1 TO 11
45 PRINT
50 NEXT Z
55 CALL SCREEN(2)
65 REM -- MARQUEF
70 LET A$="FFFFFFFFFFFFFFFF"
80 CALL CHAR(96,A$)
81 CALL HCHAR(6,6,96,22)
82 CALL HCHAR(16,6,96,22)
83 CALL VCHAR(6,5,96,11)
84 CALL VCHAR(6,28,96,11)
90 FOR HUE=3 TO 15
95 CALL COLOR(9,HUE,HUF)
100 CALL COLOR(5,HUE+1,2)
101 CALL COLOR(6,HUE+1,2)
102 CALL COLOR(7,HUE+1,2)
110 FOR DELAY=1 TO 150
120 NEXT DELAY
130 NEXT HUE
140 GOTO 90
```

Assignment Question 17-3

```
10 REM HUMPTY DUMPTY IS SQUARE
15 CALL CLEAR
20 CALL SCREEN(2)
25 REM -- MAKE A SQUARE CHARACTER
26 CALL CHAR(42,"FFFFFFFFFFFFFFFF")
27 REM -- 16 LETTER F'S
30 FOR C=3 TO 14
35 CALL COLOR(2,C,2)
40 FOR J=1 TO 28 50 REM -- ERASE THE OLD SQUARE
51 CALL VCHAR(C*2-5,J,32)
55 REM -- DRAW THE NEW SQUARE BELOW
56 CALL VCHAR(C*2-5,J+1,42)
57 FOR DELAY=1 TO 35
58 NEXT DELAY
60 NEXT J
70 NEXT C
80 GOTO 15
```

Assignment_Question_17-4

```
1 REM -- SINBAD'S MAGIC CARPET
2 GOTO 1000
198 REM
199 REM --MAIN LOOP
200 REM
210 FOR I=I TO 11
211 FOR J=I TO 11
213 K=I+J-1
214 C=C+8
218 IF C<159 THEN 220
219 C=42
220 CALL HCHAR(I,K+5,C)
221 CALL HCHAR(K,I+5,C)
222 CALL HCHAR(23-I,K+5,C)
223 CALL HCHAR(23-K,I+5,C)
224 CALL HCHAR(K,28-I,C)
225 CALL HCHAR(I,28-K,C)
226 CALL HCHAR(23-I,28-K,C)
227 CALL HCHAR(23-K,28-I,C)
228 CALL VCHAR(23-I,28-K,C)
229 CALL VCHAR(23-K,28-I,C)
290 NEXT J
291 NEXT I
999 GOTO 999
1000 REM
1001 REM -- SINBAD'S MAGIC CARPET
1002 REM
2000 CALL CLEAR
2001 CALL SCREEN(2)
3000 FOR I=2 TO 16
3005 CALL COLOR(I,I,I)
3010 NEXT I
3200 C=42
3990 CALL CLEAR
3999 GOTO 200
```

Next month we will introduce you to the DATA, READ and RESTORE statements but in the meantime please make sure that you understand fully the answers to Lesson 17 above and what makes them work the way they do.

Bye for now.

END OF ARTICLE

Compiling C

by Geoff Trott

I am in email contact with Tony McGovern as he is at the University of Newcastle and I am at the University of Wollongong. He very kindly sends me files that he has received from his wide contacts around the world that he thinks I may be interested in. One of those packages that he sent me last year was an upgrade to c99 which included 80 column access. This came from Winfried Winkler in Berlin, Germany. It is an upgrade on the original c99 from Clint Pulley of Ontario, Canada. This interested me because not only was it an upgrade to the C compiler but all the sources for the compiler were included along with those for the CSUP and CFIO support files. This means that problems can be looked at locally, rather than relying on getting upgrades, but the program file version of the compiler has to be generated. This caused me some head scratching along the way, and I thought you might be interested in the problems encountered and how they were overcome.

The files in the package included fourteen C source files, five assembler source files and three documentation files. One of the files appears twice, once as a C source file and then as an assembler source of the same functions, optimised by hand. There are two utility programs supplied, apart from the compiler. One is an optimiser, for reducing the size of the assembler source produced by the compiler, and the other is a utility to remove out of range errors on jumps by putting in branches (B instead of JMP) instead. One of the options in the new compiler is to use JMPs rather than Bs which leads to a shorter program but may lead to range errors which then need to be fixed.

According to the documentation, this version of the compiler (version 5.0, July 1993) is the first update since the last one by Clint Pulley (version 4.0, October 1987) and does the following: general clean up of code (re-name, re-group, re-write to use all compiler features); improves code for && and || operators, incrementing and decrementing characters, do...while loops; corrects error with comma operator in conditionals (? : ,); extends support for 'goto' statement in inner loops; implements Branch to Jump conversion in the compiler itself; added full support for \ escapes in string and char constants; added support for strings longer than one input line via \; suppresses EVEN in case of already even addresses; revised CSUP and CFIO not to use or destroy R3, R4, R5, R6; requires Editor Assembler sub-routines to be added to program files by hand; added support for \f and \t characters in CSUP; added 80 column version of CSUP (needs and loads CHARA1); removed C$STRT, C$END in CSUP (use S$STRT, S$END instead); added CURPOS (cursor address) variable and BL @C$SCRL (scroll screen) sub-routine definition to CSUP; CSUP uses Editor Assembler loader code buffer for scrolling; changed implementation of GPL linkage.

I am not sure if that list of improvements means anything to you as there are only a few that I felt were important to me. However I thought it would be good to have the latest version for the few programs I write. To get a C program to run, first the program must be written and saved in a file using a program editor. I use Funnelweb and use ";C" on the end of the filename to signify a C source file. This file must be compiled and the compiler will produce a file in assembler source code. I use ";S" on the end of the same filename to indicate that this is the assembler source file generated from the C source file. Then I get this ";S" filename as the workfile in Funnelweb and run it through the assembler. The advantage of making the ";S" file the workfile is that no filenames need be typed into the assembler as it will just change the last "S" into an "O" for the output of the assembler using Funnelweb. The next step is to load the object files into memory which can be most easily done with the Scriptloader from Funnelweb. This is a way of automating Editor Assembler Option 3 loading and running. Once the program is running correctly, and if it is to be used frequently, it can be saved in program format by defining SFIRST, SLOAD and SLAST (using C99PFI and C99PFF) and loading in the SAVE utility (FSAVE is probably better). This may seem to be a long, involved process to run a program, but the production of an intermediate file in the form of the assembler source allows this file to be edited to generate smaller programs, for example.

In the case of the compiler itself, I first must generate the new compiler using the version 4.0 compiler and then generate the compiler again using itself to make sure there were no problems and to make use of all the new facilities in the new compiler. The documentation that came with the package (as a file) said to first assemble CC_SBR, CSUP80, and CFIO which were three of the assembler source files. The last two of these are "library" files for the C language, providing basic support for input and output and files. The first were C compiler subroutines. Then the two utility programs OPTIM and J2B were compiled and assembled. I found one problem here with a line of text giving an error as it went over three lines. It was done for the version 5 compiler but I had to divide it up into multiple write statements for the version 4 compiler. Then OPTIM required functions called match() and nmatch() for recognising when strings were the same or not, which I did not have in my version of STRINGS. I did have string compare functions which were easily amended to become the match functions and then the programs both loaded and ran with the new CSUP80 and CFIO files. I set up script files to run both of these utilities. The next step was to compile the C compiler itself.

This was done, and generated a file of 360 odd sectors called CC;S. This used 11 of the C source files in the package as input via the #include statement. This output file was run through the optimiser which reported that it had reduced the memory requirements by more than 2000 bytes. When I tried to load the result, I obtained memory full errors before it really got going. This happened with the Editor Assembler module as well as Funnelweb. The documentation said that there would be "out of range errors" from the assembler but I did not receive these. I then examined the object code files with a sector editor. When I looked at the assembled file before the optimiser, I found that the size of the relocatable block was >6000 bytes or 24 Kbytes. This is too big as the total memory block is only that big. After optimising, the size dropped to >5772 but the other files were of size >130 (CC_SBR), >482 (CSUP80) and >400 (CFIO). Also, the main program uses some absolute code for storing data at >8330 to >8349, >20BA to >20E5, and >FD00 to >FFD1. I edited the assembler source to stop data being actually stored in the absolute areas and then assembled the optimised program. I was then able to load the programs but it would not run. It would start, but when I put in the source file name and it went off to open the file, it went into never-never land. After a bit of thought, some discussion with Rolf, and some reading of the Editor Assembler manual, I decided on a different strategy.

It was obvious that memory was very tight and that until I got the new compiler working and producing code which used branches (B) everywhere instead of jumps (JMP) (a saving of two bytes each .time), it would be difficult to load the program into the normal memory. I was using a SuperSpace cartridge which has an extra 8 Kbytes of memory so what I needed to do was make use of this extra memory. The first thing I did was to change the addresses where the data was stored from the >2000 area and >FD00 area into the >6000 area. Then the compiler was compiled, optimised and assembled. Then I wrote a small assembler program which used absolute addressing to change the contents of >2024 to >6500 and of >2026 to >7FFF. These two words hold the values of first address and last address of the memory used for the relocatable area for the loader. This was loaded after the compiler program to change the addresses to point to the cartridge memory space for the rest of the programs. Doing this allowed the compiler to load and run so that I was then able to compile all the programs again with the version 5 compiler. Now when the program was optimised and assembled, there were a number of "out of range errors" so that I then had to pass the program through the J2B program to change those jumps back to branches. On assembling the source again, there was still one error which then required one more pass through the J2B program. The size of the optimised program was >4DD4 which increased to >4F0C after application of the J2B program. There are a large number of jumps generated by the compiler.

After all this it worked. I was able to save a program version of the program which took 3 files. It needed the editor assembler cartridge or Funnelweb to load as the utilities are required in low memory, but I do not think that is a problem these days. I need to make program files of the optimiser and the jump to branch converter programs as well as these will be used in the future. Now I have an 80 column version of the compiler I shall have to write some more useful programs. I usually write them in BASIC to get the logic correct and then convert them to C for the speed.

END OF ARTICLE

Lets talk about computers

By Robert Brown

Welcome to a NEW series of articles. Yes - I can not believe it either! Since we have a few new members coming into the club, I thought I would write some articles about computing in general. The brians behind the keyboard, the guts of the machine... you get the idea. Ross Mudie has also had this idea (Did he steal it from me... or was it the other way... anyway...) This first article concentrates on the idea of binary numbering and also hex octal numbering systems. Lets get the show going......

Computer CPUs, the contents of memory and other storage devices, and computer peripherals (printers, etc) work by switching electric signals and/or circuits ON and OFF. If OFF is represented by the number 0 and ON by the number 1, then computer circuits and their electric currents can be used to count in binary (or base 2) arithmetic. The term bit, which is used to represent the smallest unit of data held by a computer (a 1 or a 0), is made from the words BInary digiT.

The binary digits, 0 and 1, therefore represent the ON/OFF signals inside a computer circuit. For this reason, all instructions sent are in binary. It is possible to represent any number, no matter how large, by using the binary digits 0 and 1.

While computers work in binary, humans generally do not. So binary values are often turned into other forms to make them easier for us to use. The two most common forms used are Octal (base 8) and Hexadecimal (base 16). Both are used instead of Decimal (base 10) because they are more closely related to the binary number system i.e. Octal 8 = 2 3 (that is 2 to the power of 3, or 2 x 2 x 2 = 8) and Hexadecimal 16 = 2 4, while Decimal 10 = 2 3.3219281.

In the binary counting system only two different digits exist, 0 and 1. The octal system has eight digits; 0, 1, 2, 3, 4, 5, 6 and 7. The hexadecimal system has 16 different digits, six more than the decimal system and six more than we have normal numerical symbols for. The extra symbols used are the first six letters of the alphabet. So hexadecimal counting goes from 0 to 9. like decimal, and then A, B, C, D, E and F.

Binary, octal and hexadecimal counting is no different from decimal counting. The same rules apply to all systems. They only differ in the number of different symbols that they use to represent their values. Once the last symbol in a column has been reached, that column is reset to 0 and 1 is added to the next column on the left. This happens when you go from 9 to 10 in decimal, or from 17 (pronounced "one-seven") to 20 ("two-zero") in octal, or from 3F ("three-eff") to 40 ("four-zero") in hexadecimal.

The table below gives values in decimal, binary, octal and hexadecimal.
(See Table 1)

Each digit column in all number systems represents powers of the base number. In the familiar decimal system, for example, the digit columns go by powers of 10. Column 0 is the right-most (or units) digit position.
(SeeTable 2)

Conversions from binary, octal and hexadecimal to decimal can be made by the following expansions:-
(See Table 3)

Converting from decimal to the other systems involves dividing repeatedly by the new base number until 0 is the answer. The remainders, in reverse order, form the new base value.
(See Table 4)

Conversions between binary, octal and hexadecimal systems are even easier:-
(See Table 5)

Bits, Nibbles, Bytes, Words:

A bit (from the words BInary digiT) is the smallest unit of data that the CPU can handle. As its name implies, a bit is a single binary digit and can be either 0 or 1. The term nibble$mw**to four bits. A nibble can therefore store a number from 0000 to 1111 binary, or 0 to 15 decimal, or 0 to F hex.

The term byte is commonly used for indicating memory or storage capacity in computers, floppy and hard disks, etc. A byte is 8 bits or 2 nibbles - (think in terms of a nibble being only a small "bite"). So a byte can hold

a number from 00000000 to 11111111 binary, or 0 to 255 decimal, or 0 to FF hex. The more common unit of memory is the kilobyte or "k" (e.g. 64k = 64 kilobytes). Because of the way a byte is defined (8 bits), 1k is not 1,000 bytes but is based on the calculation 1k = 2 10 which is 1,024 decimal bytes.

The bits in a byte are numbered from 7, the MSB (Most Significant Bit) on the left, to 0, the LSB (Least Significant Bit) on the right.

```
bit 7 6 5 4 3 2 1 0
     MSB           LSB
```

A word is made of two or more bytes joined together. The size of the words used by a particular CPU depends upon how many data bits it can process at one time.

8 bit CPUs (like the Z80 and 6502) use 8 bit (1 byte) units for their data inputs and outputs so they have data words of 1 byte in length. Internally, many 8 bit CPUs can handle 16 bit (2 byte) data words for calculations, etc. All 8 bit CPUs are able to address their memory in 16 bit (2 byte) units. This gives them direct access to a maximum of 64k (that's 11111111 11111111 binary or FFFF hex or 65,536 decimal) of memory. Computer systems based on these two CPUs frequently come supplied with 128k or more of memory but only 64k can be directly addressed by the CPU at any one time.

The TI is a 16 bit machine. The Address Bus works at 16 bit - this is the output from the main processor ie TMS9900. The Data Bus is also 16 bit ie communications between the TMS9900 and the fast RAM is done in 2 byte words. All other input and/or output is done in 8 bits.

16 and 32 bit CPUs (like the 8088, 80286, 68000, etc) can process larger data chunks and directly address more memory because they can handle much larger word lengths.

Well, that really sums up binary, hex and octal. Basically I do not have anything else to say on the matter, except that you don't really have to learn how to convert numbers between say hex deciminal, as we have calculators to do it for us. Well, it is nice to know anyway.

Next month, we look at computer programming languages in general and what languages are available. Also don't forget to look for that new article on GAMES... written by the one and only.....

```
1000010 1011001 1000101 (Binary)               Answer


 42      59      45      (Hex)                   Next
 101     114     114     (Octal)                 Month!!!!
```

(Table 1)
The table below gives values in decimal, binary, octal and hexadecimal.

Decimal	Binary	Octal	Hex	Decimal	Binary	Octal	Hex
0	00000000	000	00	10	00001010	012	0A
1	00000001	001	01	11	00001011	013	0B
2	00000010	002	02	12	00001100	014	0C
3	00000011	003	03	13	00001101	015	0D
4	00000100	004	04	14	00001110	016	0E
5	00000101	005	05	15	00001111	017	0F
6	00000110	006	06	16	00010000	020	10
7	00000111	007	07	32	00100000	040	20
8	00001000	010	08	64	01000000	100	40
9	00001001	011	09	128	10000000	200	80

(Table 2)

	Decimal	Hexadecimal	Binary
Column 0	10 0 = 1 x	16 0 = 1 x	2 0 = 1 x
Column 1	10 1 = 10 x	16 1 = 16 x	2 1 = 2 x
Column 2	10 2 = 100 x	16 2 = 256 x	2 2 = 4 x
Column 3	10 3 = 1,000 x	16 3 = 4,096 x	2 3 = 8 x
Column 4	10 4 = 10,000 x	16 4 = 65,536 x	2 4 = 16 x
Column 5	10 5 = 100,000 x	16 5 = 1,048,576 x	2 5 = 32 x
	etc	etc	etc

(Table 3)

```
+-----------------------------------------------------------------------------+
| Converting BINARY           Converting OCTAL     Converting HEXADECIMAL     |
|   to decimal                  to decimal            to decimal              |
|                                                                             |
|  01110101                   165                   75                        |
|  ||||||||+---x1    =   1    ||+-----x1  -  5      |+-----x1   -   5         |
|  |||||||+----x2    =   0    |+------x8  - 48      +------x16  - 112         |
|  ||||||+-----x4    =   4    +-------x64 - 64                                |
|  |||||+------x8    =   8                                    117             |
|  ||||+-------x16   =  16                 117                                |
|  |||+--------x32   =  32                                                    |
|  ||+---------x64   =  64                                                    |
|  |+--------- x128  =   0                                                    |
|                                                                             |
|                      117                                                   |
+-----------------------------------------------------------------------------+
```

(Table 4)

```
+-------------------------------------+-------------------------------------+
| Number  Divide by     Remainder     | Number  Divide by     Remainder     |
|          2 to give                  |          8 to give                  |
|-------------------------------------+-------------------------------------|
|  106       53          0 --------+  |   223      27           7     +     |
|   +--------+                     |  |    +-------+                  |     |
|   53       26          1 ------+ |  |    27       3           3    +|     |
|   +--------+                   | |  |    +-------+                 ||     |
|   26       13          0 -----+| |  |     3       0           3   +||     |
|   +--------+                  || |  |                                     |
|   13        6          1 ---+ || |  |     223 decimal = octal 337         |
|   +--------+                | || |  |-------------------------------------|
|    6        3          0 --+| || |  | Number  Divide by     Remainder     |
|   +--------+               || || |  |          16 to give                 |
|    3        1          1 -+|| || |  |                                     |
|   +--------+              ||| || |  |   469      29           5     +     |
|    1        0          1 -+||||||  |    +-------+                  |     |
|                                    |    29       1          13     |     |
|   106 decimal = binary 1101010     |    +-------+          = D   +|     |
|                                    |     1       0           1   +||     |
+-------------------------------------+-------------------------------------+
```

(Table 5)

binary to octal	group binary digits into 3s (from the right) then convert. It may be necessary to add one or two 0s to the front to complete a trio of digits. 1011001 = 001 011 001 1 3 1 = 131 octal
binary to hex	as above but group binary digits into 4s adding any necessary 0s at the front. 11100101011 = 0111 0010 1011 7 2 B = 72B hex
hex to binary	convert each hexadecimal digit to four digit binary. hex 5C3 = 5 C 3 0101 1100 0011 = 010111000011 bin
octal to binary	convert each octal digit to three digit binary. oct 715 = 7 1 5 111 001 101 = 111001101 bin

END OF ARTICLE

TI BASE TIPS
by Bob Relyea

Greetings fellow TIers. I hope there are a few of you still out there that actually read these articles and use them. I was given the job of organising the students lockers at the school where I teach, so I thought that it was a good time to refresh my memory on the value of that good TI-Base software and put it to use on the locker situation.

I used the TI-Base software for this job because it is really good for ease of entry of data and for using the sort option to help me get everything the way I wanted. I used several fields in the set-up of the school lockers-

SURNAME NAME CLASS YEAR FEES_PAID LOCKER_NO KEY_NO

When I finished entering all of the data I sorted it in various ways, depending of what information I wanted to look up. The 'sorts' that I found useful were:

1. YEAR - SURNAME , 2. SURNAME - NAME,

3. LOCKER_NO - SURNAME, 4. KEY_NO - CLASS

When I finished this all up I saved it to disk as a DV/80 file so I could call it up with Funnelweb and put a title on it, etc. Although there is a 'Display All' feature with TI Base, it is a lot easier to view the and use the material on your file with the word processor, especially if you have the 80-Column facility. If you do not, Geoff is 'hot on the trial' with his nice piece of software so you will probably be able to get it later in the year when he has it perfected. With our TI system of software, it is easy to change a Multiplan or TI Base file to the DV/80 file. It is also easy to change a DV/80 file to TI Base (described later), so the only tricky part is changing either a DV/80 file or a TI Base file to a Multiplan file. That is the (only bit) of bad news. The good news in this regard is that it is possible and you can also change any file to a Multiplan file, even from a working Basic or Extended Basic program. I am working on this at the moment so it will have to be the subject of a future article. Now, back to TI Base. What follows are a few tips that I have for you that I found helpful to know about.

1. Creating a DV/80 file with a TI Base file. To do this, while in command mode, you type in-

   SET DATDISK = DSK1.filename

   Then, type in 'Print All' and instead of going off to your printer, it creates a DV/80 file on DSK1 all ready for Funnelweb. Simple, isn't it?

2. You may have noticed when TI Base is booted up that the 'status' line has many pre-set parameters, such as the DATDISK. If they are not what you want, then you have to re-set them everytime you want to use the program, which can be a pain. To permanently set them the way you like, use the following steps:

   * Type in 'MODIFY COMMAND SETUP'
   * Use Fctn 4 to insert lines & type in your parameters, such as-

   SET DATDISK=DSK2

   When finished entering all your parameters, use Fctn 8 (with the write protector off the disk) to have saved to disk. N.B. This is only saved to disk at this stage so you will not note a change until the next time you load up the software. To have it changed for the current session change it the way you normally do from the command line.

3. Have you ever noticed that after doing a 'sort' that the Record Numbers (RECNUM) are jumbled? This is because the Record Numbers are just re-assigned to the line that the record that they were associated to was placed in the sort file. It is possible to have the sorted file with Record Numbers numbered in ascending order from 1 to whatever by following the following procedure:

   a) Right after doing a sort, from the command line type in MODIFY STRUCTURE and as soon as the structure screen comes up type in Fctn 8 which brings up another screen:

   ABORT - leave as it was
   RESTORE data from old structure
   USE data as is

   You choose the RESTORE choice and after the disk drives whirl away for awhile and does their bit, you can call up the file that you sorted (display all) and find that the numbering is now in ascending order but the file is still sorted. That is the way I like it!

4. Finally, a word about converting files to TI-Base files. I tried this recently with a DV/80 file and had instant success. You have to do a bit of ground work to get it to come out the way you want, however. If fact, you have to do exactly the same thing that you would do if you were going to sort a DV/80 file with TI-SORT. Incidentally, converting a DV/80 file to TI-Base is only useful for fixed records, that is, ones that have been created using (for example) the TAB function where each record is in the same place on each line. Even though it will convert any DV/80 file, it is only sensible to do it for fixed records. To have work out right you have to count the number of columns (with of field) for each field you want and note it down on paper. Make sure that it totals 80 so that it exactly matches the TI-Base system. Then, boot up TI-Base and type in-

   CONVERT (input file) (output file) [Go]

The 'Go' only has to be typed in if you have to change disks, in other words, if you do not have two disk drives. The instructions in the manual on page 3-27 are clear (for a change!) and there are several other things that you have to do after typing in the above. The first thing is to type in all your fields and widths, etc when they are asked for using the exact same format that you use when you are creating a file from scratch with TI-Base. The rest of the instructions are clear enough not to need repeat them here. I made a DV/80 file out of one of the sorted locker files mentioned above and then converted it back to TI-Base for practice. There is a bonus in doing this as I noticed that the record numbers assigned to each record was in ascending order- just the way I like it! In spite of the fact that the locker file had 200 entries it converted it fairly quickly. A Multiplan file could be converted to TI-Base the same way and with the same speed.

Well, that is all for now folks.

END OF ARTICLE

MODULE REVIEW

From Stephen Shaw in England

MODULE REVIEW- 4A FLYER- GAME-DATABIOTIC
The author was told to write a flight simulator within a tight time limit, in machine code, to fit entirely within 8k. Under the circumstances it is probably not a bad job, but it is instruments ONLY, and if you point your plane at the ground, there comes a point where you reach a maximum speed even with the throttle wide open - and it is not that fast a speed. There are various anomolies and the instrument only flying gets rather boring quite quickly. For very little more (in dollar terms anyway) you could have the so very much superior Spad XIII, so get that one and enjoy some scenery!

MODULE REVIEW- FROGGER- PARKER BROS.
This module is an excellent reproduction of the very popular (well, some years ago...) classic arcade game. It plays very much like the arcade machine. What more praise can there be! If you like Frogger, buy it.
HMMM... dont remember Frogger eh! You must guide a frog over several lanes of busy road without being squashed, THEN across a river, without falling in, being eaten by crocs, or dumped by submerging turtles... and so on.

MODULE REVIEW- D STATION I- DATABIOTICS.
This started off life as a disk program from the Bethany based International User Group, and is a VERY simple program, although well written. From a fixed base at screen centre you must shoot down flyers and parachutists. Thats about it. Graphics are extremely barren but the game can be difficult. You aim by rotating your gun (180 degree capability) which makes a change from moving left and right at screen bottom.

MODULE REVIEW- DEFENDER- ATARISOFT.
This is possibly the best Atarisoft module, and is a first rate emulation of the classic arcade game, with a small wide range radar scan at screen top showing where all aliens are, and a playing area to move your ship around while being attacked from all sides from all sorts. The play can become really frantic. This module should be in your game collection. Supplies are generally good but you may need to pay a slight premium price.

MODULE REVIEW- TI INVADERS- TI MODULE
A quick check of the museum piece arcadeart off a little too slowly fov machines shows that they too start very slowly and take a while to speed up- this does NOT make it easy to clear the first screen incidentally!
This TI Module is almost certainly the very best version of Invaders on ANY home computer. The graphics are if anything better than on the arcade machines, and all the arcade play is there.
This is one of only two modules that TI officially released on disk- the disk version had a TEST mode not in the module which allowed you to really slow the game down, and to start at any level UP TO LEVEL 50 (and it went on after level 50 too).
Another module which you really should have in your collection if you dont have it. You may not play it too much, but it is a lovely example of what TI could do. Not hard to find.

MODULE REVIEW- JUMPY - TI MODULE
Possibly the ONLY TI Module actually WRITTEN on a TI99/4A system- the others were developed and assembled on larger 9900 computers. You control a cute little Kangaroo who pushes boxes around trying to trap monsters- trapped monsters die and score points. The game becomes surprisingly difficult at level ten. Nice game, fairly difficult, good game action. Reasonable graphics.

MODULE REVIEW- MS PACMAN- ATARISOFT.
Remarkably like PacMan... Ms PacMan if you forgot is the one with the ribbons! This version has a faster screen set up (PacMan as a module for the TI had a very slow screen set up!) and more varied screen layouts. Also introducing the bouncing strawberry -so much more difficult to collect than a static bonus fruit. You eat dots and avoid monsters in a maze, clear all the dots for next maze, go over power points to temporarily gain the strength to shoo the monsters back home.
Better than PacMan, comparable to MunchMan, which seems to play faster.

MODULE REVIEW- MUNCHMAN -TI MODULE
Way back when, when PacMan was all the rage and PacMan Fever failed to make the pop charts, TI were thinking of modules for their new Home Computer. A variant of PacMan came out, and Atari jumped on it hard. Gobbling dots was OUT! So a quick rewrite and we instead eat up a trail while the requisite monsters chase us around the maze! This one can get incredibly fast, and you need some pretty good joysticks to play well, keep trying.

MODULE REVIEW: MUNCHMAN II. A Triton/Databiotics module. 1987.
Source: Try offering Database a suitable bribe to import some or try TexComp. Supplies scarce.

It has been brought to my attention that some TI owners-especially disk owners- have entirely the wrong impression of what this module is! You will see from the above review that the original Munchman module was actually version 2 - which makes this one Version 3!
This is a complete rewrite by (former?) TI employee John Phillips, responsible for so much module work. Here he has married the two connected screens of Sewermania with the first Munchman variant in which instead of laying a trail you remove one! Yes there is a screen to the right of the first one which you enter through two passages- OR you can use a roving teleport machine! And you must clear BOTH screens to go on! The hoonos are FASTER than you are so you cannot outrun them - except after going over a power dot when you can outrun them and gain good points! You must use skill and tactics in this version. A worthwhile addition to any collection of PacMan variants!

END OF ARTICLE

SHAKE-OUT IN THE HARDWARE WARS

Retyped by Loren West

A fierce pricing battle claims a major casualty

At olympic sales in downtown Los Angeles last week, an Atari home computer that cost $630:00 three years ago carried a price tag of $77:95. At Lechmere Sales in Cambridge, Mass, Texas Instruments micros that retailed for $525:00 in 1981 could be had for less than $100:00. Gemco stores in California were selling Commodore 64 computers for $199:00 each, two-thirds off their price of six months ago. In Chicago, K mart was unloading tiny Timex Sinclair 1000s, last year at $99:95, for $29:97 each.

The market for the smallist computers, always competitive, has finally blossomed into a full-scale price war. Manufacturers have been trumpeting price cuts and rebates and spendind heavily on TV ads. Until recently the high-pressure strategy seemed to be paying off, sales of 5 million home computers were predicted for 1983, a huge increase over last year's record-breaking 2.5 million.

Lately however, there have been signs that the market may be approaching saturation. Consumers are beginning to complain that with out expensive printers and disc drives, many of the low-priced machines are little more than video-game players with built-in keyboards. Talmis Inc, an Oak Park,Ill, market research firm, estimates that small computers have been selling at a monthly rate of 275,000, but manufacturers have been shipping more than 450,000 a month.

Last week the fragile market cracked. Reacting to news that Texas Instruments, suffering from disappointing hardware and software sales, had predicted a $100 million loss in its current quarter, Wall Street turned negative on the company's stock Shares of TI, one of the world's largest manufacturers of silicon chips dropped 40 points in one day, trimming nearly $1 billion from the company's paper value. On the heels of Atari's multimillion dollar loss last quarter, it looked as if one segment of the computer revolution was wobbling.

Fortunately for the indrustry, higher priced microcomputers have not succumbed to the price cutting that has bedeviled the bottom portion of the market. Apple and IBM continue to sell full price personal computers ($1,500 to $4,000) as fast as they can ship them. Even among the low-end companies there have occasionally been flashes of rational pricing strategy. Timex, for example, has systematically reduced its prices on the Timex Sinclair 1000 to help clear the way for more powerfull and more expensive models due later this year.

Not so Commodore and Texas Instruments. Says William Bowman of Spinnaker Software, a leading publisher of home computer software. "They aren't responding to the market but to each other." The Commodore - TI donnybrook began last August, when Texas Instruments offered a $100:00 rebate on the TI99/4A, bringing the machine's price down to $149:00, Commodore answered with a $50:00 trim on its VIC 20, making its tag $149:00. Then as Christmas approached, Commodore sliced an additional $20:00 off the VIC 20 price. Then $30:00 more. "The bottom line," says Bowman, "is that the VIC was cheaper than the TI during the height of the Chrismas season." Commodore Founder Jack Tramiel, a veteran of the pocket calculator wars of the 1970s adopted the slogen "Microprocessors for the masses" and boasted that he had sold more tha 1 million VIC 20s in the 1982, nearly double the sales claimed by his nearest competitors, Timex and TI.

Tramiel moved again early this year, cutting prices on his Model 64 computer and offering rebates for trade-ins on his comptitors machines. He had also pulled out of indepedent computer stores and concentrated on mass market outlets such as K mart and Toys R Us. TI machines, hobbled by second rate software, suffered a further blow in February, when the company disclosed that users were vulnerable to possible electrical shocks. TI corrected the defect, but the damage to the machine's image was not so easily repaired.

Many analysts believe that an industry shake-out is due. Atari, trying to stage a comeback with a series of lightweight, foreign-built models, remains beset by shipping delays and management turmoil. Timex, having cut back production on its model 1000, will face tough competition in the under $200:00 market for its new line. Tandy, whose Radio Shack TRS-80 Colour Computer sold sluggishly at $399:95, has dropped its price to $199:95, a level some observers believe is still not low enough to ensure the machines survival. Says Talmis President Jeanne Dietsch. "They seem to bring in new products with total disregard for their competitors' pricing."

More complications are on the horizon. Earlier this month Coleco, a vigorous competitor of Atari in the video-game business, introduced a $600:00 home computer and word-processing system that includes a typewriter quality printer. Coleco's entry, named Adam, may signal the spread of the price war from basic computer units to plug-in peripheral devices. And last week a Consortium of 14 Japanese manufacturers, led by NEC and Matsushita, announced a technical agreement that will result in interchangeable game cartridges and programs for their home computers. The new standards were developed by Microsoft, a programming house based in Bellevue, Wash., that designed the software specifications for IBM when that giant firm entered the medium price personal computer market two years ago. IBM, interestingly, is quietly working on its own low-end machines developed under the code name "Peanut." While it is not clear which standards IBM's machine will follow, one thing is certain, whichever way IBM goes, a good portion of the industry is sure to follow.

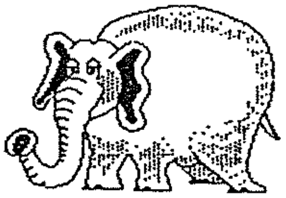
From the editor.
This article was given to me recently by Kari Leskinen, who thought that other members would be interested in reading about our trusty TI.
I would like to point out that this article is dated 1983

END OF ARTICLE

ACCOUNTANCY

```
10 CALL CLEAR
15 DISPLAY AT(3,6):"*ACCOUNT
ANCY QUIZ*"
20 DISPLAY AT(5,14):"BY"
25 DISPLAY AT(7,8):"VINCENT
MAKER"
30 DISPLAY AT(10,13):"FOR"
35 DISPLAY AT(12,12):"SCHERI
"
40 INPUT "PRESS ENTER TO GO
ON":A$
45 RIGHT=0
50 WRONG=0
55 CALL CLEAR
60 DISPLAY AT(3,5):"1.WHICH
IS THE ACCOUNTANCY    EQUATI
ON?"
65 PRINT
```

```
70 DISPLAY AT(8,3):"A)A=P-L
                    B)E=I+C
                    C)50
                     D)A=1/2
BH"
75 GOSUB 260
80 IF J=65 THEN 95
85 GOSUB 285
90 GOTO 100
95 GOSUB 315
100 DISPLAY AT(3,5):"COMPLET
E THE SENTENCE""DEBIT THE CR
EDITOR AND CREDIT THE...  "
105 DISPLAY AT(9,3):"A)CLERK
                     B)DEBTO
R                    C)ACCOU
NTANT                D)GOVT"
110 GOSUB 260
115 IF J=66 THFN 130
120 GOSUB 285
125 GOTO 135
130 GOSUB 315
135 DISPLAY AT(3,5):"IN A TR
IAL BALANCE WHAT     COMES B
EFORE ""LEDGFRS""?"
140 DISPLAY AT(8,3):"A)JOURN
ALS                  B)LEDGE
RETTES               C)B ALA
NCESHEET             D)NOTHI
NG"
145 GOSUB 260
150 IF J=65 THEN 165
155 GOSUB 285
160 GOTO 170
165 GOSUB 315
170 DISPLAY AT(3,5):"4.WHAT
IS AN END OF BUSINESS (NOT C
OMPANY) CALLED?"
175 DISPLAY AT(8,3):"A)TERMI
NATION               B)ENDIN
G                    C)DISSO
LUTION               D)NOTHI
NG"
180 GOSUB 260
185 IF J=67 THFN 200
190 GOSUB 285
195 GOTO 205
200 GOSUB 315
205 INPUT "PRESS ENTER TO SE
E WHAT YOUR GRADE IS...":F$
210 IF RIGHT=4 THEN A$="EXCE
LLENT!!...4/4....GRADE A"
215 IF RIGHT=3 THEN A$="VERY
GOOD...3/4...GRADE B"
220 IF RIGHT=2 THEN A$="FAIR
225 IF RIGHT<2 THEN A$="HOPE
LESS....LESS THAN TWO CORREC
T"
230 CALL CLEAR
235 PRINT "YOU GOT ";RIGHT;"
RIGHT AND ";WRONG;" WRONG."
240 PRINT
245 PRINT
250 PRINT A$
255 STOP
260 PRINT "PICK A LETTER..."
265 CALL KEY(0,J,K)
270 IF K=0 THEN 265
275 PRINT
280 RETURN
285 PRINT "INCORRFCT"
290 WRONG=WRONG+1
295 PRINT
300 INPUT "ENTFR TO GO ON":L
$
305 CALL CLFAR
310 RETURN
315 PRINT "CORRFCT"
320 RIGHT=RIGHT+1
325 PRINT
330 INPUT "FNTFR TO GO ON.."
:L$
335 CALL CLEAR
340 RETURN
```



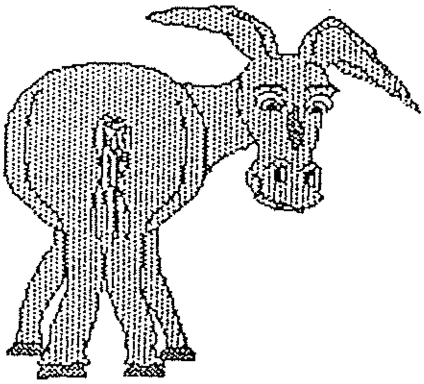
TYPING FOR ACCURACY

I found this programme on one of my favourite disks amongst other games. The disk was marked "Best of 99ER".
It is a typing game for all ages and for those of us who use all of two pointer fingers, if you wish to increase your typing skill and show your other fingers what they should be doing then type in this programme and give it a go. (ED)

```
100 REM * TYPING FOR ACCURAC
Y *
110 REM
120 DIM A$(20)
130 Y=20
140 DATA 15,FAD,A.AS,DAD,AD,
SAD,LAD,FALL,ALFALFA,SASS,LA
SS,DADS,LADS.FALLS,FADS
150 DATA 16,HAD,HAS,GAS,SAG,
HALL,HALLS,LADS,SAGS,HAG,LAG
,LAGS,SLAG,SHALL,SASH,DASH,F
LASH
160 DATA 18,DEAF,DEED,SEE,FE
ED,HEED,LIKE,KILL,FILL,FEEL,
FEES,LIED,DIAL,SLIDE,FLIES,L
IFE,SLID,GLIDE
170 DATA 17,TAG,HAT,TALL.THY
,DAY,HAY,JAY,GAY,LAY,TAR,RAT
,STAR,STAFF,FAST,TRY,SAY,YAR
D
180 DATA 20,WISH,EXAM.EXACT,
TEXT,TWO,WON,SOW,WASH,WORLD,
OWE,WORD,LOOK,LOSE,SOD,SOS,F
OW,TEXAS,OXEN,MIX.WORSE
190 DATA 16,QUAKE,QUIZ,QUIP,
ZAP,QUIT,PIQUE,PLAQUE,PUZZLE
,PLAZA,SAP,ZIPPER,PRIZE,QUIC
K,SQUEEZE,ZEAL,ZIP
200 DATA 18,CALM,CAN,MEN,NIM
BLE,EXACT,EXAM,MIX,NIX,BUZZ,
ZOOM,NAVY,CAB,BACK,BOMB,ZOMB
IE,CAVE,VACATE,VARMINT,
210 GOSUB 920
220 GOSUB 1540
230 GOSUB 1690
240 CALL KEY(0,KEY,STS)
250 IF KEY<49 THEN 240
260 IF K>56 THEN 240
270 ON (KEY-48)GOTO 280.300.
3r0,340,360,380,400.910
280 RESTORE 140
290 GOTO 410
300 RESTORE 150
310 GOTO 410
320 RESTORE 160
330 GOTO 410
340 RESTORE 170
350 GOTO 410
360 RESTORE 180
370 GOTO 410
380 RESTORE 190
390 GDTO 410
400 RESTORE 200
410 CALL CLEAR
420 CALL COLOR(1,2,1)
430 CALL SCREEN(8)
440 GOSUB 1890
450 READ N
```

```
460 FOR I=1 TO N
470 READ A$(I)
480 NEXT I
490 SCORE=0
500 RANDOMIZE
510 FOR K=1 TO 10
520 W=INT(N*RND)+1
530 IF A$(W)="0" THEN 520
540 XC=24-K
550 CALL SOUND(1500,-1,2)
560 FOR J=1 TO LEN(A$(W))
570 CALL HCHAR(XC,J+17,ASC(S
EGS(A$(W),J,1)))
580 NEXT J
590 INPUT B$
600 IF B$=A$(W)THEN 630
610 CALL SOUND(1000,-7,1)
620 GOTO 650
630 CALL SOUND(1000,-2,1)
640 SCORE=SCORE+1
650 IF K<>1 THEN 710
660 CALL HCHAR(23,Y-4,98)
670 CALL HCHAR(23,Y+4,98)
680 CALL HCHAR(23,Y-3,99)
690 CALL HCHAR(23,Y+3,100)
700 GOTO 740
710 IF K<>2 THEN 740
720 CALL HCHAR(23,Y-4,99)
730 CALL HCHAR(23,Y+4,100)
740 CALL HCHAR(23,Y-1,105,3)
750 CALL SOUND(1,44000,30)
760 CALL HCHAR(XC-1,Y-2,98,7
)
770 CALL HCHAR(XC-1,Y+5,32,3
)
780 CALL HCHAR(23,1,32,15)
790 A$(W)="0"
800 NEXT K
810 CALL CLEAR
820 FOR I=2 TO 8
830 CALL COLOR(I,2,1)
840 NEXT I
850 SC=10*SCORE
860 PRINT : : : : :"YOUR SCO
RE IS";SC,"PERCENT ACCURACY.
"
870 PRINT : : :
880 CALL KEY(0,KEY,ST)
890 IF KEY<>13 THEN 880
900 GOTO 230
910 END
920 CALL CLEAR
930 CALL SCREEN(5)
940 T=500
950 CALL SOUND(T,880,3,698,8
,294,10)
960 PRINT : : :TAB(11);"T Y
P I N G"
970 CALL CHAR(104,"FFFF00FFF
F00FFFF")
980 CALL SOUND(T,932,3,784,8
,196,11)
990 PRINT : :TAB(15);"FOR"
1000 CALL COLOR(10,16,6)
1010 CALL COLOR(11,13,1)
1020 CALL SOUND(T,784,3,659,
8,262,10)
1030 PRINT : :TAB(12);"ACCUR
ACY"
1040 CALL CHAR(112,"00E0F8FE
FFFFFFFF")
1050 CALL SOUND(T,880,3,698,
8;175,12)
1060 CALL CHAR(113,"00000000
80E0F8FE")
1070 CALL CHAR(114,"80E0FCFF
FFFCE08")
1080 CALL CHAR(115,"FEF8E08"
)
1090 CALL SOUND(T,698,3,587,
8,233,10)
1100 PRINT : : : : : : :
1110 PRINT : : :
1120 CALL CHAR(116,"FFFFFFFF
FEF8E")
1130 CALL SOUND(T/2,784,3,16
5,10)
1140 CALL CHAR(96,"80C0E0F8F
8E0C08")
1150 CALL SOUND(T/2,698,3,16
5,10)
1160 CALL CHAR(97,"80C0E0F0F
8FCFEFF")
1170 CALL SOUND(T/2,659,3,27
7,10)
1180 CALL CHAR(98,"FFFFFFFFF
FFFFFFF")
1190 CALL CHAR(117,"FFFFFFFF
FFFFFFFF")
1200 CALL SOUND(T/2,784,3,27
7,10)
1210 CALL CHAR(99,"FFFFFCF8F
0E0C08")
1220 CALL SOUND(T*2,698,2,58
7,8,147,12)
1230 CALL HCHAR(15,7,117,7)
1240 CALL HCHAR(16,7,117,9)
1250 CALL HCHAR(17,7,117,7)
1260 CALL HCHAR(16,16,114)
1270 CALL HCHAR(15,15,113)
1280 CALL HCHAR(17,15,115)
1290 CALL HCHAR(15,14,112)
1300 CALL HCHAR(17,14,116)
1310 CALL SOUND(T,466,4,165,
10)
1320 FOR XX=15 TO 17
1330 CALL HCHAR(XX,1,104,6)
1340 NEXT XX
1350 CALL SOUND(T,440,3,175,
10)
1360 CALL COLOR(2,16,1)
1370 FOR YY=18 TO 26 STEP 2
1380 CALL HCHAR(16,YY,42)
1390 NEXT YY
1400 CALL SOUND(T,698,3,440,
8,294,10)
1410 CALL HCHAR(16,28,49)
1420 CALL HCHAR(16,29,48,2)
1430 CALL HCHAR(16,31,37)
1440 CALL SOUND(T,784,3,587,
8,233,10)
1450 CALL CHAR(100,"7F3F1F0F
070301")
1460 CALL CHAR(101,"00010307
0F1F3F7F")
1470 CALL SOUND(T/2,698,3,39
2,8,262,10)
1480 CALL CHAR(102,"00001818
3C3C7EFF")
1490 CALL SOUND(T/2,659,2,26
2,10)
1500 CALL CHAR(105,"DBDBDBDB
DBDBDBDB")
1510 CALL SOUND(4*T,698,2,44
0,8,175,10)
1520 CALL SOUND(1,44000,30)
1530 RETURN
1540 CALL CLEAR
1550 CALL SCREEN(7)
1560 CALL COLOR(2,2,1)
1570 PRINT "PICK A TYPING CA
TEGORY."
1580 PRINT : : :"YOU WILL SE
E A WORD"
1590 PRINT "IN THE ROCKET."
1600 PRINT :"TYPE AND ENTER
IT BEFORE THE TONE ENDS."
1610 PRINT :"IF YOU ARE CORR
ECT,"
```

```
1620 PRINT "ANOTHER TONE SOU
NDS,"
1630 PRINT :"IF YOU ARE INCO
RRECT."
1640 PRINT "YOU WILL BE BLAS
TED.": : : :
1650 PRINT "PRESS ENTER TO C
ONTINUE."
1660 CALL KEY(0,KEY,ST)
1670 IF KEY<>13 THEN 1660
1680 RETURN
1690 CALL CLEAR
1700 CALL SCREEN(12)
1710 CALL COLOR(1,2,12)
1720 FOR I=2 TO 8
1730 CALL COLOR(I,1,12)
1740 NEXT I
1750 PRINT : : : :" CHOOSE O
NE": :
1760 PRINT :" 1 HOME KEYS"
1770 PRINT :" 2 HOME ROW"
1780 PRINT :" 3 TOP ROW, MID
DLE FINGER"
1790 PRINT :" 4 TOP ROW, POI
NTER FINGER"
1800 PRINT :" 5 RING FINGER"
1810 PRINT :" 6 LITTLE FINGE
R"
1820 PRINT :" 7 BOTTOM ROW"
1830 PRINT :" 8 END PROGRAM"
: : :
1840 CALL SCREEN(5)
1850 FOR I=2 TO 8
1860 CALL COLOR(I,2,12)
1870 NEXT I
1880 RETURN
1890 CALL COLOR(9,1,1)
1900 CALL COLOR(10,1,1)
1910 CALL VCHAR(12,Y,98,13)
1920 CALL VCHAR(13,Y-1,98,12
)
1930 CALL VCHAR(13,Y+1,98,12
)
1940 CALL VCHAR(14,Y-2,98,11
)
1950 CALL VCHAR(14,Y+2,98,11
)
1960 CALL VCHAR(13,Y-2,101)
1970 CALL VCHAR(13,Y+2,97)
1980 CALL VCHAR(12,Y-1,101)
1990 CALL VCHAR(12,Y+1,97)
2000 CALL VCHAR(11,Y,102)
2010 CALL VCHAR(22,Y-3,98,3)
2020 CALL VCHAR(22,Y+3,98,3)
2030 CALL VCHAR(23,Y-4,98,2)
2040 CALL VCHAR(23,Y+4,98,2)
2050 CALL VCHAR(22,Y-4,101)
2060 CALL VCHAR(22,Y+4,97)
2070 CALL VCHAR(21,Y-3,101)
2080 CALL VCHAR(21,Y+3,97)
2090 CALL COLOR(9,7,1)
2100 CALL COLOR(10,16,1)
2110 RETURN
2120 END
```



OUTDENTING and INDENTING
WORD PROCESSING TIPS
by Bob Relyea

N.B. In this article the following abbreviations are used:

LF = linefeed CR = carriage return

I used to admire how some authors could make each first word of a sentence in a document be placed exactly where it should be without a lot of untidy-looking overlapping. To my shame, I never thoroughly looked into it until recently when I was writing up the minutes of the recent AGM. Prior to that I just tidied up a document manually by using the fixed mode. So, I said to myself- "Right! I am going to get this document right by using formatting commands if I have to sit here all night." So, with an occasional look into the TI-WRITER manual, a couple of questions to a knowledgeable individual, and a lot of practice, I finally got it right!

If I may digress for a moment, I was grateful that my mini-system has FunnelWeb on eproms as it made it easy to go back and forth between Editor and Formatter to see how it looked. As you may know, you do not have to send a formatted text off to a printer. You can format it to disk when it asks you for 'print device' name and then call it up like any other file and have a look at it. This is called 'preview' (or paper-saver) in the PC world, and if you have FunnelWeb on eproms it is just as fast to execute. This method eliminates the problem of sending a text to a printer only to discover that something did not come out the way you wanted it to and must be done again. While I am on the topic, if you do format something to disk and call it up for a look then you will notice that there is a linefeed (LF) symbol at the end of each line, even a blank line. So, if you want to print this formatted text right then and there off the editor then you must put a linefeed symbol at the end of your printer parameters to avoid double spacing. The reason is that the editor already has a linefeed build into it and with the linefeeds on the document you get two lots. One, therefore, must be suppressed and, as indicated above, this is done by typing in something like the following after invoking the PF (printfile command):

```
PIO.LF     or
RS232.BA=4800.LF
```

Before I get back to the original topic, a couple more thoughts. If you have used either the @ or the & in the text to overstrike (embold) or to underline then you will notice on the formatted text that something like the following will result:

```
overstrike          underline
overstrike          _________LF
overstrike
overstrikeLF
```

Any word that is overstruck will appear four (4) times in the formatted, only the last of which will have a linefeed. So, the printer will go over the same space four times before being instructed after the fourth one to linefeed. That is why is becomes darker- the same word has been printed four times in the same place. For the underline there appears only the word (with no linefeed) and the underline itself is on the next line with a linefeed. So, after printing the word, the printer does not advance but goes right back over the same place with an underline before it advances to the next line. Of all the things in the Word Processor bag of tricks, I use this feature the most. Finally, if you format a text to disk, you can keep the same name if it is sent off to a drive other than the one that the original text is on. If not, then you must change the name as it will not format onto itself.

Now, back to the original topic. I am going to illustrate what I mean by having the first word 'in the right place' by using an example out of the index of a Nuclear Physics text book. I will place the usual formatting commands at the beginning so you can see the result. Later I will show what must be done if you are to get it to look like it does in the book.

Example 1:

```
Chapter 1   Early Observations of Radioactivity
            1.1 The discovery of radioactivity
            1.2 The ionising ability of radioactivity
            1.3 Other radioactive elements
            1.4 Exercise

Chapter 2   Types of radioactive emissions
            2.1 The main emissions
            2.2 The nature of radioactive emissions
            2.3 The relationship between penetrating and
                ionising ability
            2.4 Exercise
```

See how nice and neat the text appears with everything lined up perfectly? That is the way that it appears in the book and what you would get printed if you typed it in the editor and just printed it without going through the formatter. However, many times we want it to pass through the formatter, and if we do so with the above text with the formatting commands as given below what follows is what you get:

```
.FI;AD;LM0;RM55;IN+5CR

     Chapter 1 Early Observations of Radioactivity
     1.1 The discovery of radioactivity
     1.2 The ionising ability of radioactivity
     1.3 Other radioactive elements
     1.4 Exercise

     Chapter 2 Types of radioactive emissions
     2.1 The main emissions
     2.2 The nature of radioactive emissions
     2.3    The    relationship    between    penetrating    and
ionising ability

     2.4 Exercise
```

Pretty yukky isn't it? Well, the main point of this article is to explain how to pass the above text through the formatter and get the original text all lined up nicely. Outdenting and indenting is explained, in brief, on pages 54, 103, and 143 of the TI Writer manual, if you have one. The INDENT (.IN) command is used with the LEFT MARGIN (.LM) command. In the manual it states:

LEFT MARGIN = .LM n (.LM [+:-]n)- This command sets the left margin. The "n" in the command is replaced with the column number at which the left margin is to be set. This number must be smaller than the number at which the right margin is set. The left margin may be reset with this command as many times as desired.

INDENT = .IN n (.IN [+:-]n)- This command indents the first lines of paragraphs. The Text Formatter interprets the beginning of a paragraph as the first character after a carriage return symbol or the first character after a blank line or lines. When an absolute value is used (i.e. without a + or -), text is indented to that column number regardless of the left margin. When relative values are used, the indent point is calculated from the left margin value. For example, if the left margin is 10, .IN +5 sets the indent point at 10 + 5, or 15 and .IN-5 set the indent point at 10-5, or 5. The command .IN +0 can be used to nullify the Indent command, but another Indent command must be inserted where indenting is to resume. Each time the left margin is reset, indentation must also be reset.

Normally then, the Indent command is used in conjunction with the Left Margin command. So, to set up proper commands to indent and outdent attention must be given to both the Indent and Left Margin numbers. Indent, of course means that the first line of the paragraph is set to the right of the normal Left Margin, and is normally achieved by using the + sign. Outdent means that the first line of the paragraph is set to the left of the normal Left Margin and is normally achieved by using the - sign. The emphasis of the following discussion will be on Outdenting.

The 'trick' in understanding outdenting is to realise that, in a paragraph, the Left Margin deals mainly with the second line onwards, while the Indent command deals exclusively with the first line. Remember, a paragraph ends with a carriage return symbol no matter how long or short the 'paragraph' is. So, in the above text that we are dealing with, there would normally be a carriage return symbol at the end of each part of the outline, which virtually means at the end of each line.

Let's look at the first three lines of the text we are working with. To get this right you must do a bit of counting. You may also have to consider that, when passed through the formatter, there are at least two (2) spaces placed after each full stop and at least one (space) after all other symbols or letters. So, if we want the following text to end up correctly, we notice that there are twelve (12) spaces from the 'C' of chapter to the 'E' of Early. This means that there must be 12 to accompany the Outdenting command. The Outdent comand must be twelve (12) to the left of the left Margin. So, your document may be set as follows:

```
.LM12;IN-12CR
Chapter 1   Early Observations of Radioactivity
            1.1 The discovery of radioactivity
            1.2 The ionising ability of radioactivity
```

Notice I said 'may be' set out. It only really matters that the Indent command is -12 and that the Left Margin is less than the right one and is at least 12 in this case. So, I could have written in .LM20;IN-12 and the result would have been basically the same. The only difference is that in the second case the whole text will be 8 units further to the right on the page than in the first case.

The last thing to consider is what to do if a point in the outline is longer than one line. If care is not taken you will get an ugly-looking overlap.

Let's consider the second part of the outline. Notice that point 2.3 cannot be fitted onto one line. The following two examples show firstly what the result should be and secondly what will happen if you do not set the commands properly.

```
Chapter 2   Types of radioactive emissions
            2.1 The main emissions
            2.2 The nature of radioactive emissions
            2.3 The relationship between penetrating and
                ionising ability
            2.4 Exercise

Chapter 2   Types of radioactive emissions
            2.1 The main emissions
            2.2 The nature of radioactive emissions
            2.3  The  relationship  between  penetrating
            and ionising ability
            2.4 Exercise
```

Notice that second line in point 2.3 starts directly under number 2.3, which appears untidy to me. One way around it is to place the commands as follows:

```
.LM12;IN-12CR
Chapter 2  Types of radioactive emissions
.LM16;IN-4CR
            2.1 The main emissions
            2.2 The nature of radioactive emissions
            2.3 The relationship between penetrating and
                ionising ability
            2.4 Exercise
```

A quick bit of counting will explain why. You will notice that all of the words following the numbers 2.1, etc are at Left Margin 16 and that the numbers are 4 units to the left of them. Need I say more? That is what outdenting is all about. You just set the Left Margin at 16 and outdent -4.

The next thing is to point out that because of the Adjust command that usually accompanies a formatted text, you ocassionally notice more than two spaces after a full stop and more than one space after other symbols/words. The following is an illustration of what I mean. Keep your eyes on the words after numbers 2.3.

```
.LM12;IN-12CR
Chapter 2  Types of radioactive emissions
.LM16;IN-4CR
            2.1 The main emissions
            2.2 The nature of radioactive emissions
            2.3  The  relationship  between  penetrating
                and ionising ability
            2.4 Exercise
```

Notice how the 'T' of The is one space further to the right than you probably wanted? To get around this you just place a carot(^) symbol between words that you do not want to be separated. The following is an illustration:

```
.LM12;IN-12CR
Chapter^2^^^Types of radioactive emissions
.LM16;IN-4CR
            2.1^The main emissions
            2.2^The nature of radioactive emissions
            2.3^The relationship between penetrating and
                ionising ability
            2.4^Exercise
```

Now, the last thing that I wanted to mention is more of a general use for indenting. It is common to place the majority of the formatting commands at the beginning of a text, before a title even. What follows is a typical example:

```
    .FI;AD;LM0;RM55;PL330;IN+5
```

If you place the .IN here then the title (s) will be placed further to the right than you wanted and it is not really centered at all. The reason is that the Indent command places the first line of any paragraph 5 units (in this case) to the right as it counts from the Left Margin. Since every line of a title is the first line of a paragraph it results with a title too far to the right. To get around this, place the Indent after the title but before the main text and it will come out better. Cheers.

END OF ARTICLE

JUST A ONE LINER (ED)

Joe: " I once had a parrot for five years and it never said a word."
Bill: "It must have been tongue-tied."
Joe: "No, it was stuffed.

TI - TILER

Written by
Col Christensen
17 Centaur Street
REDCLIFFE AUST 4020

This program is shareware and may be freely distributed by any means at all except that a fee for the program itself must not be charged. If you like TI-TILER and hope to make sufficient use of it in the future, then a contribution towards the time and effort put into producing it would be appreciated. For convenience, any remittances from USA could be in the form of notes as they can be easily converted to Australian Dollars and used (if I ever get enough) to order software etc from the States. The program is available through the TIsHUG club shop.

TI-TILER provides the facility of producing neat electronics circuit diagrams or a variety of graphs on the TI-99/4A, saving and loading to and from disk and finally emerging as a hardcopy through a printer.

TI-Tiler executes quickly in all modes. This speed is achieved by using "tiles", predefined characters, as the building blocks for the circuit diagrams.

The program is not limited to producing electronic circuitry or graphs only, but can be adapted for many other drawing needs .by using a set of tiles suited to the purpose. Other sets of tiles may be created in due course to suit other needs.

TI-Tiler creates designs that can take up a full page of A4 paper. Lines are made up of 60 tiles across the page (approx 8 inches) and there can be 90 lines of tiles down the page (approx 10 inches). The screen acts as a window to part of the full page, each screen being 30 tiles wide and 18 tiles deep.

Window changing takes place like in TI-Writer except that in "Roll up" and "Roll down", like "Next Screen", one window overlaps the next by half a screen. The position of the window on the page is indicated on the screen by a ruler line across the bottom and another down the right hand side of the screen.

The program loads using the filenames TT and TU through any memory image loader. If loaded through the F'web loader or from one of the titles on the F'web option screen, TI-Tiler will keep the screen colours of F'web and on quitting by using TI's option 6, will return to F'web.

The title screen options are:-
1. DRAW on design in memory
2. LOAD a design or tiles
3. SAVE a design to disk
4. CLEAR design from memory
5. DUMP design to printer
6. DIRECTORY lising
7. QUIT after saving design

In creating a design by selecting option 1 (DRAW) there are three screen modes. The top line on the screen is the Mode Line and shows status and/or help info. The interconnecting paths to the various modes is depicted here.

```
      TITLE SCREEN
           |
      OPTION SCREEN
           |
       DRAW MODE--------TILE MODE
           |
       TEXT MODE
```

Use of the [FCTN/9] key takes you BACK

DRAW MODE

The screen mode line shows:-

DRAW - 3 Joyst XSPBTC,.Ovrlay

In the draw mode, the joystick is used to move the cursor around the page. On the mode line at the top of the screen the symbol following the word "DRAW" is the tile that has been loaded for printing. Pressing the fire button will deposit this tile on the relevant part of the page governed by the cursor position. You can move the cursor in any direction while holding the fire button to make rows of tiles.

The "3" on the status line at the top of the screen indicates the joystick cursor repeat speed. Speed 3 is a good average speed.

Following the "3" is a list of keys etc that are active in the draw mode.
[B] is used to blank out a tile under the cursor. This can only be done when the joystick is centralised and there is no movement of the cursor.

[D] Pressing this key will pick up whatever tile is under the cursor for DUPLICATION whenever the fire button is pressed.

[C] steps through a list of screen COLOUR combinations so that you can select the best one for the resolution of both the TV screen and your eyes.

[,] Pressing the comma or LESS key changes the repeat speed of the joystick cursor to a slower rate.

[.] The full-stop or MORE key increases the speed of the joystick cursor.

[S] At the bottom of the screen is a set of tiles that can be used. Pressing the S key will SWAP this tile set for another. See below for description of tiles.

[X] The X key with the downward arrow on it takes you down to select a tile from those at the bottom of the screen. See the TILES section below.

[T] The T key takes you to the TEXT mode. See later description of this mode.

"Overlay" shows that you could refer to the key overlay to find the uses of the keys across the top of the keyboard. (See later section OVERLAY.) In the draw mode pressing one of the keys from 1 to = at the top of the keyboard automatically loads one of the most commonly used tiles for line drawing.

[FCTN/9] returns you to the option selection screen.

TILES MODE-

The screen mode line shows:-

TILES S Joystick Fire

You get to TILES mode from the draw mode by pressing the [X] key (the down arrow on it serves as a prompt). In this mode you select the tile you wish to use by placing the cursor over it and pressing the fire button. Pressing the fire button takes you straight back into the DRAW mode to the position where you left.

While in the tile mode pressing the S key will SWAP the tile sets the same as it does in the draw mode.

TEXT MODE-

The screen mode line shows:-

TEXT Arrow Overlay CTRL/123CM

This mode has a close resemblance to many of TI-Writer's features. All ASCII keys are functional as well as the 4 arrow keys for cursor movement. In addition, The OVERLAY (See later section) shows that this mode provides (through the FCTN key combined with the top row of keyboard keys) the functions of insert char, delete char, insert line, delete line, roll up, roll down, next screen and escape similar to TI-Writer.

TI-TILER provides three oddball functions. FCTN/7 rotates the ASCII character set 90 degrees clockwise which is useful for making keyboard overlays and long circuits that will only fit down the page. To toggle the characters back to the upright position just press FCTN/7 again.
If you wish to dump your design to printer with horizontal characters, you must press [FCTN/9] to leave the TEXT mode while the characters are rotated. The program will temporarily restore them to normal until either re-entering the TEXT mode again or during the actual process of printing the design. In other words, the position of the text in the TEXT mode is the way they will be DUMPed to printer.

FCTN/0 sets the auto cursor movement direction downwards to facilitate typing vertically. After typing this way, it is good practice to reset immediately to horizontal typing by pressing FCTN/=. (Yes, the Quit combination has been disabled)

The [ENTER] key is active and will return the cursor to the position of the first tile at the beginning of the next line. If vertical typing is invoked through the FCTN/0 keypress, the cursor will be returned to the top tile in the column to the left of the one it was in before. In vertical typing the top of page is assumed to be the right hand side of the paper. One way to HOME the cursor is to press [ENTER], change the typing direction and press [ENTER] again.

In the text mode, the COPY and MOVE features are available through the control key. Sections of the design in the shape of a rectangle can be copied or moved. Use the arrow keys to :-
Move the cursor to the top-left corner of the required rectangle and press CTRL/1.
Move the cursor to the proposed bottom-right corner and press CTRL/2. The rectangle (square if you like) has now been marked.
Move the cursor to the top-left corner of the destination position and press CTRL/3.
Now you can copy or move it by pressing CTRL/C or CTRL/M but, for saftey, you must first verify your action before the actual copy or move is executed. If the section you wish to copy/move won't fit in the destination, the computer will respond with a "honk". The three co-ordinates are kept in memory until changed. To make multuple copies of part of your design to different parts of the page, all you need to change for each is the destination.

LOADING FROM DISK-

Option 2 allows the loading of -designs- or -tiles- stored on a floppy disk, ramdisk or hard drive DSK1. All tile and design filenames have a maximum of 8 user characters followed by the suffix "_T" for easy identification on a disk directory.

You can differentiate between the two on a directory because a design file uses 23 sectors of disk space while a tile set uses a much smaller number, 9 sectors for a full set from character 0 to 253. When entering the filename to load do not include the suffix. See also the section on disk storage protocols for TILES below.
[FCTN/9] allows you to return to the DRAW mode and pressing it a second time takes you to the option selection screen.

SAVING TO DISK-

The filename for the design you have created can be up to 8 characters long. The program will automatically add the _T suffix.

CLEARING A DESIGN FROM MEMORY-

Just as the option indicates, this is the way to wipe the page clean before starting a new design. If this option is selected, you still must confirm your intentions before the command is executed.

DUMP TO PRINTER-

This option will dump the page contents in memory to PIO. NOTE, if you want the text printed in the rotated position, you must have set the characters to this position when leaving the TEXT mode. Before printing, you are asked to set the number of printer overstrikes per line. Just use 1 for a draft copy, but for the final version the weaker the colour of your printer ribbon, the higher the number of overstrikes to use. Printing although not slow takes a little while. For each line printed, the computer and printer must handle 960 bytes of code. Each of the 480 dots across the page of 60 tiles is printed twice to improve the appearance of the printed product.

If your printer has a devicename different from PIO.LF.CR, it is necessary to patch the program code using a sector editor such as Disk-Edit. The program code needs the full name of the printer such as RS232.BA=9600.LF.CR as well as a name length byte to precede it.

Code in the TT file that governs the printer line spacing is ESC "3" 24 and will be found during a disk sector search for 1B3318. If your printer doesn't recognise this code then the ESC "0" code could be used to replace it. Converted to hexadecimal you would replace the 1B3318 with 1B3000. You must include the 00 at the end as the program sends 3 bytes to the printer and the printer will understand only the 1B30 code and should ignore any null byte following it.

DIRECTORY LISTING-

After selecting the directory option you need to input the drive number to read from. If the list of filenames is too large, just those that will fill fit on the screen will first be shown. A keypress will allow the listing to be continued.

QUIT-

The normal quit key has been disabled for obvious reasons so this option is a way out when you have finished.

If you had loaded TI-Tiler through F'web, you will be returned to that program otherwise there will be a software reset to the colour bar screen.

THE TILES-

The tiles are staggered chessboard style on the screen for clarity. The first set seen when the program is first loaded begin with the line drawing tiles. This set comprises character numbers 128 to 187 in an order shown here:

```
128    132    136    140    144 etc
  129    133    137    141    145
130    134    138    142    146
  131    135    139    143    147
```

The second set shown when the key [S] is pressed in the DRAW mode extends from character number 188 to 247. Characters 248 and 249 are redefined to make the ruler lines on the screen but the 6 remaining characters are not used.

The third set shown are characters 0 to 59. From 0 to 31 are or can be defined as tiles with the exception of the cursor, number 30. The characters shown on the right half of the screen are the lower end of the normal ASCII character set.

You can get a printout of the tile sets by loading the file "TILESETS" which is included on your TI-TILER disk, and then loading whichever tile set you want, either T1EL, T2EL or T3GR. Go BACK to the option screen and dump the design to printer.

TILE SETS-

Tile sets are assembled and stored on disk files in much the same way as TI-Writer character sets. Tiles are defined in the same format as Basic character definitions are done.

Tile set disk files can be made to cover just the text character set, just the upper characters from 128 up, just the lower chars from 0 to 31 or combinations of these. The ones supplied on disk with TI-Tiler cover the full range from 0 to 247. T1EL uses standard sized ASCII chars while T2EL and T3GR use the smaller sized TI-writer Editor ones. The "EL" sets are for drawing electronic schematic circuits and the "GR" set has tiles suited to making graphs.

The area of VDP where the tiles get loaded depends on the disk file header. This header forms the first 6 bytes of the first disk sector where the file is stored. The 6 bytes when combined to 3 words have the following meanings:

0000 Indicates this is last file.
0800 VDP address base to load to.
07C0 Number of bytes to load starting at the VDP base.

Here is a table to show the relationship between VDP addresses, the character number whose definition starts from there and where on disk this definition starts on a full tile set file such as T1EL.

VDP	Char No	Disk sector
>0800	0	1st + 6 bytes
>0900	32	2nd + 6 bytes
>0A00	64	3rd + 6 bytes
>0B00	96	4th + 6 bytes
>0C00	128	5th + 6 bytes
>0D00	160	6th + 6 bytes
>0E00	192	7th + 6 bytes
>0F00	224	8th + 6 bytes

OVERLAY-

Included on the TI-Tiler disk is a program called OL/OM and a design file called OVERLAY_T. The OL program is a patched version of the TT/TU program specially modified to print keyboard overlays of the correct size. This produces the key overlay that, when printed out can be placed above the top row of computer keys for reference.

Load up OL with an EA option #5 type loader. From the option screen, load the design file, OVERLAY and also the Tile set, T2EL, with its smaller ASCII characters. Then select the DRAW option. From there press [T] to reach the TEXT mode. Press FCTN/7 to rotate the chars 90 degrees. Press FCTN/9 twice to return to the option screen.

Select the DUMP option and when asked how many overstrikes, select 2 or 3. Your overlay will soon be printed out. Keyboard overlays for other programs can be created and printed using the OL/OM program.

Other designs and their relevant tile sets shown in brackets that are included on the TI-TILER disk for your convenience and can be printed out for reference are:

ELECDEMO (T1EL or T2EL)
GRAPHDEMO (T3GR)
TILESETS (T1EL/T2EL or T3GR)

END OF ARTICLE

THE TIGERCUB 14-DIGIT PROGRAMMABLE CALCULATOR
With 6-MEMORIES 6-WINDOWS & 34-FUNCTIONS

by Jim Peterson, Tigercub Software

This calculator program is available through the TIsHUG shop.

I always wanted a calculator with more than one memory, and a window to display the contents of each one. The computer has plenty of memory, and the monitor screen has plenty of room for windows, so I wrote a 6-memory 6-window calculator. Recently I decided to go back and upgrade that old program. By the time I got through I had a 6-memory 6-window 34-function programmable calculator with many other features. It was necessary to write this program to accept either numeric or alphabetic data and then sort it out. For this reason, it does not respond as instantly as a calculator. However, I think it does some things that few if any calculators can do.

When you boot this program, the screen displays 6 memory areas marked U through Z, and you are asked if you want to label them. That will help a great deal in keeping track of what you are using them for. The computer will force you to unlock the alpha lock and label them in lower case, which will make them stand out nicely in inverse video.

Next you are asked if you want a 14-digit display. Unlike the 8-bit PCs, the TI calculates to 14 digits of accuracy, but normally rounds them off to 10 digits for screen display. This option will display the full 14 digits, if it is not more than 9,999,999,999 or less than -9,999,999,999. You are required to depress the Alpha Lock again to answer this prompt, and it must stay depressed thereafter. Then you are asked if you want to use conventional or straight-line mode. Conventional mode is much like you would use with an ordinary calculator - you must press Enter after you input each value, but not after each function. For instance, 77 (Enter) + 81 (Enter) = . In straight-line mode you simply type 77+81= (Enter), which is a bit faster but the computer then pauses for a few seconds to decipher the input before giving the answer. If you want to enter large numbers in exponential notation, you must use the conventional mode.

To switch from one mode to the other, just enter J. The mode you are in is displayed in the upper right of the screen. Entering JJ will clear the memory labels and irretrievably clear all memories; Q will terminate the program. If you use the =, the result is simply displayed on screen, but if you use a memory name (U through Z) the result is placed in that memory and displayed in its window. For instance, 77+81X puts 158 in memory and window X. You can also enter a memory name to calculate with the value it contains. For instance, U (Enter) + 81 (Enter) V adds 81 to the value from U and puts the result in V. W+XY adds the values in W and X and puts the result in Y. U+UU would double the value of U. To poke a value into a memory, just enter a value and a memory name, such as 77 (Enter) U, or in straight-line 77U.

The four basic functions are + (located on Shift =) for adding, - (located on Shift /) for subtracting, / for dividing and * (located on Shift 8) for multiplication. All that shifting is a nuisance, especially if you are using one hand to keep track in a column of figures. To make it easier, you can use P (plus) for +, M (minus) for -, D (divide) for division, and T (times) for multiplication. The correct symbols will still appear on screen.

Other available functions are (power), % (percent), and R (root). 10 (Enter) 2 (Enter) = will give you 100, which is 10 to the power of 2. 10 (Enter) % 100 (Enter) = gives 10 which is 10% of 100. 3 (Enter) R 64 (Enter) = gives the 3rd root of 64, or 4. FCTN U, which is _, will give you the value of PI. _ (Enter) * 10 (Enter) = multiplies PI by 10.

When you enter a problem the name of the function you used, such as "addition" is highlighted in inverse video at the bottom of the screen, so you will know if you made a mistake. With this calculator, you can even enter a series of calculations. In conventional mode 67 (Enter) + 33 (Enter) / 2 (Enter) * 5 (Enter) U or in straight-line mode 67+33/2*5U (Enter) will add 33 to 67, display the result, divide by 2, display the result, multiply by 5 and put the value in U. You are limited only by the line length of 28 characters. But I said this calculator has 34 functions. Where are the other 26? TI Basic has a few other math functions, and in Appendix K of the Extended Basic Manual you will find the algorithms for 20 advanced math functions. I have no idea what those do, but I programmed them into my calculator. Here they are -

CTRL A	atn	CTRL N	inverse cotangent
CTRL B	cosine	CTRL O	hyperbolic sine
FCTN A	exponent	CTRL P	hyperbolic cosine
FCTN B	log	CTRL Q	hyperbolic tangent
CTRL E	sine in radians	CTRL R	hyperbolic secant
CTRL F	tangent	FCTN F	hyperbolic cosecant
CTRL G	secant	CTRL T	hyperbolic cotangent
CTRL H	cosecant	CTRL U	inverse hyperbolic sine
CTRL I	cotangent	CTRL V	inverse hyperbolic cosine
CTRL J	inverse sine	FCTN G	inverse hyperbolic tangent
CTRL K	inverse cosine	CTRL X	inverse hyperbolic secant
CTRL L	inverse secant	CTRL Y	inverse hyperbolic cosecant
CTRL M	inverse cosecant	FCTN W	inverse hyperbolic cotangent

To use one of these, enter a value, then a FCTN or CTRL and = or a memory name. 8 (Enter) FCTN A U will put the exponent of 8 in memory U. Be warned that entering invalid values in some of these will cause a numeric overflow or underflow and, since I have turned off ON WARNING to avoid spoiling the screen display, you will not be informed.

You do not have any use for those? Well then, you can reprogram them for any functions you do need. They are in lines 760 through 1080. Be sure to use A for the value being input, C for the result. C=A-.1*A+.06*A will return the value of A minus a 10% discount, plus a 6% sales tax. If you need additional variables, put their values in your memories and reference them in your equation, using M(1) through M(6) for memories U through Z. C=A*M(1)/M(6) will multiply A by the value in U and divide it by the value in Z. You can write multiple-statement equations, even multiple-line equations. Use J as a loop counter, @ for an internal variable; if you need other internal variables, use some that are not in the prescan list in line 110, and add them to that list. When you type the name you want displayed, use lower-case letters, and use FCTN C rather than the space bar for spacing. You can easily customize this calculator with a couple of dozen formulas for whatever field you are working in.

Sometimes you might want to total the values in all memories. Just enter & to total and display. To clear all the memories, enter C. To clear memory X, for instance, enter CX (in either mode).

E is the oops! key. Enter E to restore the last previous values in all memories, or EU, for instance, to restore the last previous value in U.

Sometimes you may just want to add up a series of numbers. Enter A= if you just want the totals displayed, or AU, for instance, to accumulate the total in U. You are now in cumulative mode and each value you enter will be added to the total. Enter Q to get out of this mode. C and E are not active in this mode, and you cannot enter memory names.

If you want a hard copy of your work, enter FCTN 0 and the memory labels, names and values will be output to your printer. If you selected the 14-digit option, the printout will also be in 14-digit format.

To save your work, enter I to save all memories, or IU, for instance, to save a specific one. You will be prompted for a disk drive/filename. To retrieve the data, enter O for all memories or OV, for instance, for a specific one.

To access the base conversion mode, enter B. You will be able to convert any number from/to any base from 2 to 36. To escape this mode, enter 0 for the value to be converted.

You find it difficult to remember all those commands? At the prompt for the first value or command, just press FCTN 7 for a Help screen. Thanks to Karl Romstedt for that one.

I said this was a programmable calculator, and I was not just referring to the fact that you could reprogram those 26 functions listed above. This calculator lets you enter an equation; the program then rewrites itself while it is running, and uses the equation to solve whatever values you give it. To get into programming mode, enter #. You will be prompted to enter a formula. This must be in the form of a valid Extended Basic statement, using A for the value to be determined and B through F, as many as you need, for the values you will be prompted to input. All math functions are supported. For instance, A=B C-INT(SQR(C)). The program pauses to tokenize your input and then prompts you for values to use for, in this instance, A, B and C. You are then prompted for a memory name in which to store and display the answer.

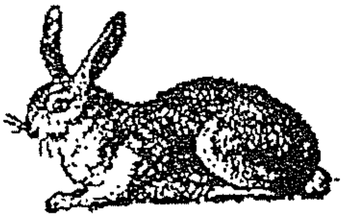
Remember the mathematical heirarchy - if you want to add or subtract before multiplying or dividing, use parentheses - A=(B-C)*D/(E-F). If your formula is not a valid XBasic statement, it will be rejected. If it is valid but incorrect for its purpose, it will give erroneous results; for instance, if you use X as a variable name, you will not be prompted for its value, which will be 0. To exit the programmed formula, enter 0 at all prompts.

Finally, this calculator contains a programmable iterative calculator to solve such difficult problems as A=B B-SQR(B), where A is the known value. These can only be solved by trial and error. To access this mode, enter @. You will be prompted for a formula, which must be in the A=B format. The computer pauses to write the equation into itself, prompts you for a value of A, and goes through a series of trial and error calculations which are displayed on screen. Then you are prompted for a memory to receive the result. To exit this mode, give A a value of 0.

I hope you find this program useful. I am releasing it to the public domain; I will not even bother to put a fairware donation request on it.

Jim Peterson

END OF ARTICLE



DATABASE
Author Unknown

This program is available through the TIsHUG club shop. Database is a program which a person can store names, addresses, phone numbers, ect....on to disk or cassette and you can retrieve them on demand. The minimum requirements for this program is a cassette recorder which makes this program great for those who just have the TI console and a cassette recorder. However, if you have a disk drive and extra memory you may utilize those devices also. The program will hold 25 people but if you have memory expansion it will hold 120.

Here is how the program works. All the information you tell the computer is stored in an array. You may change, add or delete any information you wish. When you are through you must save it all out to disk or cassette. None of the data is permenent until it is saved. When you start the program again you load the data off of disk or cassette back into the array. When you save the data onto disk the computer creates a file called "DIRECTORY" you should only have the program "DATABASE" on the disk to make sure there is enough room to store the data. If you use cassette just locate a spot on tape and record it and remember the location so you can retrieve it later. PLEASE NOTE If you have memory expansion erase line 210 and 220 and type in these two lines:

```
210 A=120
220 DIM B$(121),C(121),D$(121),E$(121),F$(121),
    G$(121),H$(121),I$(121),J$(121)
```

The program is relatively self explanatory but here are some instructions to help you through it. The first question you will be asked when you run the program is:

1. LOAD DATA FROM DISK
2. LOAD DATA FROM CASSETTE
3. INITILIZE

If this is the first time to run the program or you wish to start over select 3. If you have data stored on cassette or disk then select the appropriate choice.

Now the main selection menu will be displayed.

1. INPUT
2. RETRIEVE
3. SEARCH
4. UPDATE
5. SAVE/RE-LOAD
6. END

Select 1 if you wish to input information. Select 2 if you wish to retrieve information already in the computer. Select 3 to search for a particular person in the computer. When you select 3 to search another menu will be displayed.

1. LIST ALL NAMES IN LOG
2. LIST ALL NAMES AND ADDRESSES
3. SEARCH BY FIRST/LAST NAME
4. ABORT

If you wish to look trough all names in the computer select 1. To look through all names and addresses in the computer select 2. To search for a particular person select 3 and give either a first or last name and the computer will search until a match is found and will list all people with the name you entered. Select 4 to abort back to the main menu.

Now the main menu is back up select 4 to update information on a person. For example if a persons address or phone number etc...changes then you would use this selection to change it. When you select 4 another menu will be displayed.

1. DELETE
2. UPDATE
3. ABORT

To delete someone from the computer select 1 to delete and give the persons name. To update information on a person select 2 and re-type the information about the person. Select 3 to abort back to the main menu.

Now back on the main menu, we have the selection 5 to save or reload old information. When you select this selection another menu will be displayed.

1. SAVE TO DISK
2. SAVE TO CASSETTE
3. RE-LOAD INFORMATION
4. ABORT

The most important thing to remember is to save the information before you quit. The information is not really permenent until you save it to disk or cassette. If you wish to erase the information now in memory and re-load old information off of cassette or disk then select 3. Select 4 if you wish to abort back to the main menu.

Now back on the main menu select 5 to end the program. You will be asked if you really wish to end to make sure you did not hit that selection by accident. Just say YES if you do and NO if you do not. Remember to have saved your information if you have not then type NO and then save it.

END OF ARTICLE

LANDSCAPE PRINT

Reply-To: cohen GOMEZ.phys.virginia.edu
Organization: Dept. of Physics, University of Virginia

Here is another piece from the Freenet. Hope some of you find it useful. If you really want to print sideways from Multiplan, I would advise getting the program MP-Sideways (I think it was sold by Quality 99) or the fareware program Side*Print, they are more elaborate and will do a better job there. But for fun, the following Extended BASIC program might work out.

Make Your Printer Print Sideways!

Have you ever wished that you could print text or tables sideways on the paper (that is, with the text reading vertically instead of horizontally)? The following two programs will enable you to do this on an Epson/Gemini compatible printer. The method uses the bitmap graphics mode of the printer.

First you must create the data statements that give the printer the codes for each letter turned on its side. They cannot be typed in because they are mostly control characters. The first program creates a mergeable file for Extended BASIC for all of them. The letters created will look exactly like the ones on your screen (i.e. small letters are actually small caps) unless you put in CALL CHAR statements before line 100.

```
100 OPEN #1:"DSK3.DATAMERGE",VARIABLE 163
110 FOR X=1 TO 19 ::
    PRINT #1:CHR$(0);CHR$(5*X);CHR$(147);
120 FOR Y=1 TO 5 :: CALL CHARPAT(5*X+Y+26,C$):: GOSUB
    170 :: D$=""
130 FOR Z=8 TO 1 STEP -1 :: D$=D$&CHR$(D(Z)):: NEXT Z
140 PRINT #1:CHR$(199);CHR$(8);D$;
150 IF Y=5 THEN PRINT #1:CHR$(0)ELSE PRINT #1:CHR$(179);
160 NEXT Y :: NEXT X :: PRINT #1:CHR$(255)&CHR$(255) ::
    CLOSE #1 :: STOP
170 FOR Z=1 TO 8 :: F1$=SEG$(C$,2*Z-1,1)::
    F2$=SEG$(C$,2*Z,1)
180 F1=ASC(F1$)-48+7*(ASC(F1$)>60)
190 F2=ASC(F2$)-48+7*(ASC(F2$)>60)
200 D(Z)=16*F1+F2 :: NEXT Z :: RETURN
```

Next you type in the main program, it is actually quite short!

```
100 DIM D$(126),A$(60):: FOR X=32 TO 126 :: READ D$(X)::
    NEXT X
110 ESC$=CHR$(27):: OPEN #2:"PIO.CR" :: PRINT #2:
    ESC$&"A"&CHR$(7);ESC$;"C";CHR$(0);CHR$(11);
120 FLAG=0 :: INPUT "TEXT FILE: DSK":F$ ::

    OPEN #1:"DSK"&F$
130 FOR X=1 TO 60 :: LINPUT #1:A$(X):: IF EOF(1)THEN 150
140 NEXT X :: GOTO 170
150 FLAG=1 :: CLOSE #1 :: IF X=61 THEN 170
160 FOR X=X+1 TO 60 :: A$(X)=RPT$(" ",80):: NEXT X
170 FOR X=1 TO 60 ::
    A$(X)=A$(X)&RPT$(" ",80-LEN(A$(X))):: NEXT X :: FOR
    X=1 TO 80 :: PRINT #2:ESC$&"K"&CHR$(224)&CHR$(1)
180 FOR Y=60 TO 1 STEP -1 :: B$=SEG$(A$(Y),X,1)
190 PRINT #2:D$(ASC(B$));
200 NEXT Y :: PRINT #2:CHR$(13)&CHR$(10):: NEXT X ::
    PRINT #2:CHR$(12):: IF FLAG=0 THEN 130
210 INPUT "DO ANOTHER?(Y/N)":AN$ :: IF AN$="Y" THEN 120
    ELSE CLOSE #2
```

Note that in line 110 I have put in codes for 7/72 inch linefeeds and a formfeed of 11 inches. Check your printer codes to make sure they are the same. Now merge in the file created previously by typing MERGE DSK1.DATAMERGE This will put in 19 lines numbered from 5 to 95 in increments of 5, and if you list them they will look crazy. Do not change them!! Now save the program under whatever name you wish, and it is ready to run. The only limitation is that the file must contain only ASCII codes 32 to 126, that is no control characters or deletes (127). You can ensure this by using the PF function of the TI-Writer and typing C DSK1.yourfile. Have fun.

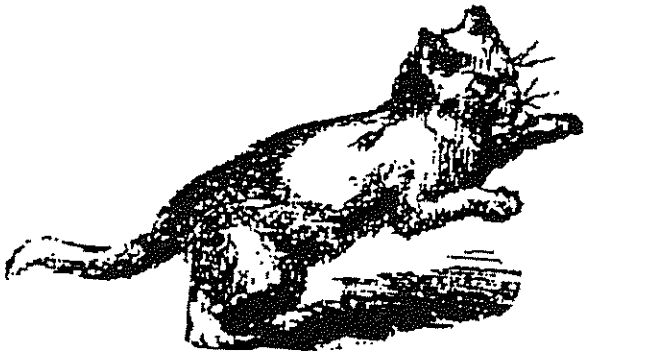
Note also that if you are sure that the right margin of your file is always less than 80, you can substitute whatever it is for 80 in lines 160 and 170 (twice). You can also use this program to print an entire Multiplan listing lengthwise by using the print on disk option and always giving a margin of 80 or less.

All you have to do is type in the program and run it.

END OF ARTICLE

FROM THE BBS

MAIL TO : ALL
MAIL FROM : GEOFF WA
SENT ON Thursday 14/07/94 at 14:38:41
Greetings from not so sunny Western australia. This is my first time on the BBS although I hope to be a frequent user, and give some news on the TI happenings here in WA.
Hope to hear from you all some time,
regards,
Geoff WARNER
SECRETARY, TIUP



TIGERCUB PRINTALL
Version 1.6.1
by Jim Peterson
****

This program is available through the TIsHUG shop. It will print your text in a choice of 1 to 5 columns, and gives you complete choice of fonts, left and right margins, spacing between columns, lines per page, etc. I think the prompts are self-explanatory.

NOTE: Some folks have thought that this program did not work because they expected it to reformat text into the desired column width. Use Reformatter+ or the FUNLWEB Formatter to do that.

It takes some time to read in text and format it into multiple columns, so if you need to print more than two copies, or will need more copies in future, it will pay you to print back to the disk. To do this, at the printer prompt type over the PIO.LF default with DSK. and a drive number and file name. The text will then be formatted and printed to a D/V 254 file.

The next prompt is for the record length, which will be the default of 80 if the text was prepared with TI-Writer or whatever. However, if you enter 254 you will be prompted for an input file name of a file printed to disk by this program, and for the number of copies wanted, which will then be printed immediately.

If you have Triton's Super Extended Basic module, you can LIST an Extended Basic program to disk in 28-column format by LIST "DSK1.filename":28:1-32766 . The result will be a D/V 28 file. With this program you can print the listing in 5 columns by selecting 28 record length, elite condensed, 5 columns, 28 column width.

This version has been modified slightly so that it will allow the use of "Control U" codes input by FUNLWEB, to underline, emphasize, double-strike, etc. an individual word or line. Note that if you are printing in more than one column you must turn off the codes at the end of the line, or they will also affect the same line in all subsequent columns. You must also remember that the control codes will be deleted in printing, which will affect the format.

If the file has a Tab setting, first enter T to get to the tab line, place a period to replace the R, then go to the end of the tab line and place an R, so that lines can be shoved right. For this reason, CTRL U codes cannot be used with full 80-column lines.

For instance, if you want to underline a word, press CTRL 0 to get the open cursor fixed mode. Position the cursor on the first letter of the word, type FCTN 2 and then space bar 3 times to open up 3 spaces, backspace to the first of these, and type CTRL U, FCTN R, CTRL U, -, CTRL U, CTRL A, CTRL U. Move the cursor to the first space beyond the word, type FCTN 2, space 3 times, backspace 3, type CTRL U, FCTN R, CTRL U, -, CTRL U, CTRL @, CTRL U.

If the word is at the end of a line or you are underlining a complete line, and the line is not completely filled with characters, go to the end of the line first and put the "turn-off" codes starting in the space just after where the last character would be. For instance, if the column width is 40, start the codes in column 41.

With this method, you can print individual lines or words in italics, double-struck, underlined, superscript, emphasized, or in different NLQ fonts or different colors. However, do not use any CTRL U codes for a feature that you plan to select from Printall, or you will turn it off for the rest of the text.

NOTE: When a line contains CTRL U codes, the program will NOT warn you or truncate a line which is more than the selected column width.

Although this program is intended primarily for multiple-column printing, it has other uses. If your letter turns out to be 70 lines long and you would like to print it on one page, use this program and select 70 lines. If you need a double-spaced manuscript, select 30 lines. If you need a tiny list, such as a list of the songs to put in the case of a music cassette, select elite condensed superscript and 120 lines per page.

Since the TI-99/4A can only store strings of about 12.5k of console memory may get a MEMORY FULL error if you try try to format much more than 60 lines of condensed print per page. You can gain an extra 1036 bytes by entering a CALL FILES(1) and then NEW before loading this program.

END OF ARTICLE

CODE

By Loren West

This programme when saved to a floppy disk under the name of "LOAD" will result in the user confronted with a screen saying that this is a "RESTRICTED GAMES DISK PLEASE ENTER CODE" after this screen a game of secret numbers will comence (high, low numbers) just work your way through the high, low numbers till you pick the right ones.
Have your favourite game or menu named "LOAD1" and when you find the two right numbers this will automatically load that programme

```
100 CALL CLEAR
110 CALL SCREEN(5)
120 INPUT "ENTER YOUR NAME ":NAME$
130 CALL CLEAR
140 CALL COLOR(2,7,12)
150 CALL HCHAR(23,3,42,28)
160 PRINT ;NAME$;"THIS IS A RESTRICTED GAMES DISK PLEASE
ENTER CODE"
170 PRINT
180 PRINT
190 PRINT
200 PRINT
210 PRINT
220 FOR A=1 TO 10
230 CALL SOUND(500,1661,2)
240 CALL SOUND(500,740,2)
250 !GOTO 27
260 NEXT A
270 ! SECRET NUMBER
280 RANDOMIZE
290 MSG1$="SECRET NUMBER IS"
300 MSG2$="YOUR TWO NUMBERS"
310 !CALL CLEAR
320 !INPUT "ENTER LIMIT?":LIMIT
330 SECRET=INT(20*RND)+1
340 CALL CLEAR
350 N=N+1
360 INPUT "LOW,HIGH GUESSES:":LOW,HIGH
370 IF LOW<>HIGH THEN 390
380 IF SECRET=LOW THEN 470
390 IF SECRET<LOW THEN 430
400 IF SECRET>HIGH THEN 450
410 PRINT MSG1$" BETWEEN":MSG2$
420 GOTO 350
430 PRINT MSG1$" LESS THAN":MSG2$
440 GOTO 350
450 PRINT MSG1$" LARGER THAN":MSG2$
460 GOTO 350
470 RUN "DSK1.LOAD1"
```

END OF ARTICLE

VINCENT'S CORNFR

DEAR FRIENDS,

WELCOME TO ANOTHER, "VINCENT'S CORNER".I'M SORRY THERF WASN'T AN EDITION FOR THE LAST COUPLE OF MONTHS BUT I HAVE BEEN SICK AND IN HOSPITAI, RFFORF THAT...RUSY WITH WORK.

TAKE CARE,

VINCENT

P.S. I HAVE A NEW HIGH SCORF FOR YOU...YAHT7FF-348.

CALENDAR PROGRAMME FOR GW BASIC

This is a listing of a calendar programme for GW Basic, it prints the calendar on screen in three neat months across the page. Below is an example of a page dump to the printer.

```
10 CLS: DEFINT A-Z
20 DIM BD(28),DA(12,7,6),DM(12),M$(12)
30 FOR T=1 TO 12: READ DM(T): NEXT
40 FOR T=1 TO 28: READ BD(T-1): NEXT
50 FOR T=1 TO 12: READ M$(T): NEXT
60 INPUT "WHAT YEAR "; Y
70 X=Y-14: IF X<0 THEN 60
80 X= Y - (INT(X/28)*28 +14)
90 IF Y/4 = INT(Y/4) THEN DM(2)=29 ELSE DM(2)=28
100 C2=1:CO=BD(X)-1
110   FOR T=1 TO 12
120 C3=C3+1:CO=CO+1:DA(T,CO,C2)=C3:IF CO=7 THEN CO=0:C2=C2+1
130 IF C3<>DM(T) THEN 120 ELSE C3=0:C2=1
140   NEXT:  PRINT
150   FOR T=1 TO 12 STEP 3
160 LOCATE,10: PRINT M$(T) TAB(37) M$(T+1) TAB(63) M$(T+2): PRINT
180 PRINT "    S  M  T  W  T  F  S" TAB(29) "  S  M  T  W  T  F  S" TAB(55)
"  S  M  T  W  T  F  S"
190     FOR TR=1 TO 6
200       FOR TRR=1 TO 3
210 LOCATE ,(TRR-1)*26+2: PRINT " ";
220         FOR TT=1 TO 7
230 D=DA(T+TRR-1,TT,TR)
240 IF D=0 THEN D$="   ": GOTO 260
250 D$=RIGHTS(" "+STR$(D),3)
260 PRINT D$;
270         NEXT
280       NEXT:  PRINT
290     NEXT
300   NEXT
310 END
320 DATA 31,28,31,30,31,30,31,31,30,31,30,31,3,4,5,7,1,2,3,5,6,7,1,3,4,5,6,1,2,3
,4,6,7,1,2,4,5,6,7,2," JANUARY ","FEBRUARY","  MARCH  ","  APRIL  ","   MAY   ","
  JUNE  ","  JULY  "," AUGUST ","SEPTEMBER","OCTOBER","NOVEMBER"
330 DATA "DECEMBER "
```

```
         JULY                     AUGUST                   SEPTEMBER

   S  M  T  W  T  F  S       S  M  T  W  T  F  S       S  M  T  W  T  F  S
                  1  2          1  2  3  4  5  6                   1  2  3
   3  4  5  6  7  8  9       7  8  9 10 11 12 13       4  5  6  7  8  9 10
  10 11 12 13 14 15 16      14 15 16 17 18 19 20      11 12 13 14 15 16 17
  17 18 19 20 21 22 23      21 22 23 24 25 26 27      18 19 20 21 22 23 24
  24 25 26 27 28 29 30      28 29 30 31               25 26 27 28 29 30
  31
        OCTOBER                  NOVEMBER                  DECEMBER

   S  M  T  W  T  F  S       S  M  T  W  T  F  S       S  M  T  W  T  F  S
                     1             1  2  3  4  5                   1  2  3
   2  3  4  5  6  7  8       6  7  8  9 10 11 12       4  5  6  7  8  9 10
   9 10 11 12 13 14 15      13 14 15 16 17 18 19      11 12 13 14 15 16 17
  16 17 18 19 20 21 22      20 21 22 23 24 25 26      18 19 20 21 22 23 24
  23 24 25 26 27 28 29      27 28 29 30               25 26 27 28 29 30 31
  30 31
--
```

REGIONAL GROUP REPORTS

Meeting Summary For AUGUST

Central Coast	13/8/94 Saratoga
Glebe	11/8/94 Glebe
Hunter Valley	14/8 21/8/94
Illawarra	09/8/94 Keiraville
Liverpool	12/8/94 Yagoona West
Sutherland	19/8/94 Jannali

CENTRAL COAST Regional Group

Regular meetings are normally held on the second Saturday of each month, 6.30pm at the home of John Goulton, 34 Mimosa Ave., Saratoga, (043) 69 3990. Contact Russell Welham (043)92 4000.

GLEBE Regional Group

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce Street, Glebe. Contact Mike Slattery, (02) 692 8162.

HUNTER VALLEY Regional Group

The Meetings are usually held on the second or third Sunday of each month at members homes starting at 3pm. Check the location with Geoff Phillips by leaving a message on (049) 428 617. Please note that the previous phone number (049) 428 176 is now used exclusively by the ZZAP BBS which also has TI support. Geoff.

ILLAWARRA Regional Group

Regular meetings are normally held on the second Tuesday of each month after the TIsHUG Sydney meeting at 7.30pm, at the home of Geoff & Heather Trott, 20 Robsons Road, Keiraville. A variety of activities accompany our meetings, including Word Processing, Spreadsheets and hardware repairs. Contact Geoff Trott on (042) 29 6629 for more information.

* LIVERPOOL Regional Group *

Regular meeting date is the Friday folling the TIshug Sydney meeting at 7.30 pm. Contact Larry Saunders (02) 644-7377 (home). Mum will let you know were I am or when I will be home.

NOTE: I will be doing relief managering at several stores, and will not be easy to reach during the day. Some of the stores I will be managering trade to 10pm/11pm/12pm and I am working up to 14 hours a day, 5 days a week.

*** ALL WELCOME ***

12th August 1994
My Place : 34 Colechin St. Yagoona West

9th September 1994
My Place : 34 Colechin St. Yagoona West

Bye for now Larry.
Liverpool Regional Co-Ordinator

SUTHERLAND Regional Group

Regular meetings are held on the third Friday of each month at the home of Peter Young, 51 Jannali Avenue, Jannali at 7.30pm. Peter Young.

TIsHUG in Sydney

Monthly meetings start promptly at 2pm (except for full day tutorials) on the first Saturday of the month that is not part of a long weekend. They are held at the MEADOWBANK PRIMARY SCHOOL, on the corner of Thistle Street and Belmore Street, Meadowbank. Cars can enter from Gale Street and park in the school grounds. Regular items include news from the directors, the publications library, the shop, and demonstrations of monthly software.

AUGUST MEETING - 6nd AUGUST

SEPT MEETING - 3rd SEPT

The cut-off dates for submitting articles to the Editor for the TND via the BBS or otherwise are:

August - 13th August

September - 10th September

These dates are all Saturdays and there is no guarantee that they will make the magazine unless they are uploaded by 6:00 pm, at the latest. Longer articles should be to hand well before the above dates to ensure there is time to edit them.

This months list of words is based around the subject of "BIRDS"

```
Y I C U O A Q V T C S N B X H H X N Q O
Q F N S D D V M H D K S Z U W H S N J E
G A X N L D X C O S J T S B D N V U O J
W K E N T T N J P F R N X V M G B U U A
R R A L Q I M E H N D O R R A H I A K F
W A Y H F K C L F O D Q S O X Z T E V T
T B L I A I Y L V B V M E E I E W J H I
B E B L A L C E M R Z M R Q L X O Y Z B
P Z I I U G A A N M U D U S X L N Z Y Y
Y A G Y T G M G S K H G N H A T A Y Y P
S T R Z V N S S C V C C B J T B H P Z Q
W E I C I N O D U A O E M S U A I Y M P
A R U L R R X R C R W U A S N D N E J O
L N O R T O L P J O P M L Y G B P N C A
L B Q A T I W S R H P R M E X F Q B W D
O F B D A C E R Z Z E H O I L K K R U E
W L B T A P A G U O Q N O X W B X C B A
A W G C A P N H L S U J I V O I K P J G
S A F A S Q G D Q C C N D F Z V K Y O L
W Y J A U B X J N R E R F W D N X B R E
```

Find these hidden words

HOW TO PLAY

In this puzzle there are (19) words somewhere, horizontally, vertically, diagonally even backwards. Find the words and place them below in the place provided.
The answers will appear in next months magazine.
GOOD LUCK!.

```
1. _________(9)  2.      __(6)  3.      (4)
4. ___(4)        5. ___ (4)     6.    __(5)
7. __(3)         8. _____(5)    9.      (5)
10.____(4)       11. ___(4)     12.    _(5)
13.________(7)   14.___ __(7)   15.    _(7)
16.________(7)   17. ___(4)     18.     (7)
19.____(4)
```

This puzzle was compiled using Ashley Lynn's programme "word puzzle" which is avaliable from the TIsHUG shop.

Last months list of words based around "Winter Activities" are,

Skiing
Chairlift
Slalom
Snowboard
Perisher
Plough
Snowmen

Tobogan
Hotdogging
Skijump
Alps
Stocks
Charlet

Iceskates
Crosscountry
Slush
Resort
Drift
Apres