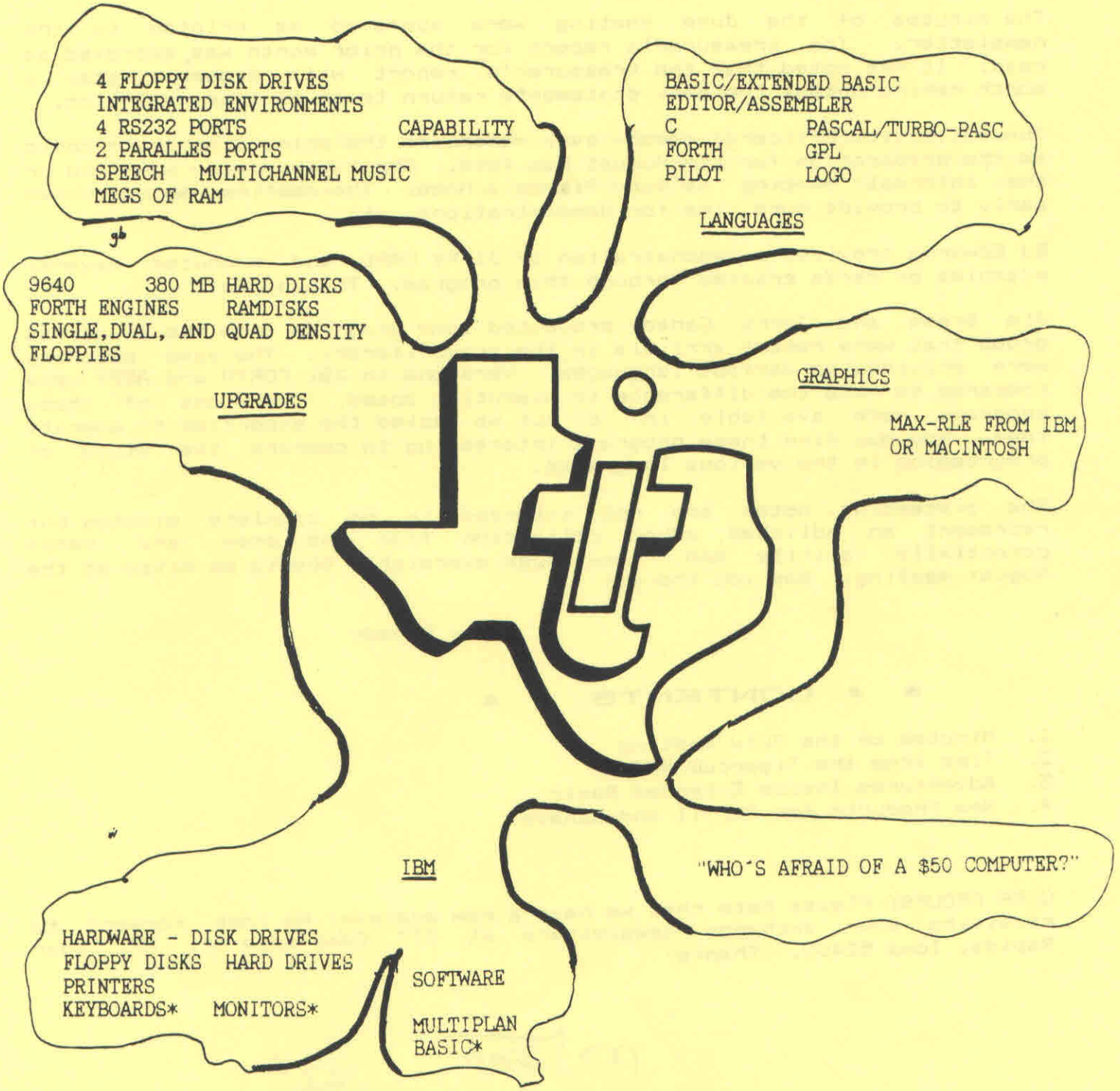
CEDAR VALLEY 99'ER USER GROUP

NEWSLETTER

CEDAR RAPIDS/MARION, IOWA

July
1989



*With adapters or appropriate hardware and/or software.

* * MINUTES FROM THE JULY MEETING * *

The July meeting was called to order by President Jerry Canady. After a period of time the absence of a secretary was noted. The attendance and other details were not formally recorded.

The minutes of the June meeting were approved as printed in the newsletter. The treasurer's report for the prior month was approved as read. It was noted that the treasurer's report will probably stay a month behind unless the bank statements return to an earlier mail slot.

The collective officers' memory bank remembers the prime discussion topic as the preparation for the August ham fest. Short discussion was held on the informal meeting at Gary Bishop's home. The meeting was adjourned early to provide more time for demonstrations, etc.

Ed Edwards provided a demonstration of JIFFY CARD and produced several examples of cards created through this program. Thanks, Ed.

Jim Green and Jerry Canady presented some programs from the Bluegrass group that were recent arrivals in the group library. The same programs were provided in various languages. Versions in XB, FORTH and ASSY were compared to note the difference in execution speed. Versions of these programs were available in 'c' but we lacked the expertise to operate them. You may find these programs interesting to compare the style of programming in the various languages.

The preceeding notes are not intended to be complete minutes but represent an adlibbed adhoc collection from the pres and veeps potentially volitile RAM. Any huge oversights should be noted at the August meeting. See you there!!

Jerry Canady

* * CONTENTS * *

1. Minutes of the July meeting
2. Tips from the Tigercub #54
3. Adventures Inside Extended Basic
4. New Products for Our TI and Geneve

USER GROUPS: Please note that we have a new address! We look forward to receiving your exchange newsletters at 377 Cambridge Dr. NE, Cedar Rapids, Iowa 52402. Thanks!

123)500

.24
500)123.00
1000
2300
2000
3000

TIPS FROM THE TIGERCUB

#54

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213

Distributed by Tigercub Software to TI-99/4A Users Groups for promotional purposes and in exchange for their newsletters. May be reprinted by non-profit users groups, with credit to Tigercub Software.

Over 120 original programs in Basic and Extended Basic, available on cassette or disk, NOW REDUCED TO JUST $1.00 EACH!, plus $1.50 per order for cassette or disk and PP&M. Minimum order of $10.00. Cassette programs will not be available after my present stock of blanks is exhausted. The Handy Dandy series, and Color Programming Tutor, are no longer available on cassette.
Descriptive catalogs, while they last, $1.00 which is deductable from your first order.

Tigercub Full Disk Collections, reduced to $5 postpaid. Each of these contains either 5 or 6 of my regular catalog programs, and the remaining disk space has been filled with some of the best public domain programs of the same category. I am NOT selling public domain programs - they are a free bonus!
TIGERCUB'S BEST, PROGRAMMING TUTOR, PROGRAMMER'S UTILITIES, BRAIN GAMES, BRAIN TEASERS, BRAIN BUSTERS!, MANEUVERING GAMES, ACTION GAMES, REFLEX AND CONCENTRATION, TWO-PLAYER GAMES, KID GAMES, MORE GAMES, WORD GAMES, ELEMENTARY MATH, MIDDLE/HIGH SCHOOL MATH, VOCABULARY AND READING, MUSICAL EDUCATION, KALEIDOSCOPES AND DISPLAYS

NUTS & BOLTS DISKS

These are full disks of 100 or more utility subprograms in MERGE format, which you can merge into your own programs and use, almost like having another hundred CALLs available in Extended Basic. Each is accompanied by printed documentation giving an example of the use of each. NUTS & BOLTS (No. 1) has 100 subprograms, a tutorial on using them, and 5 pp. documentation. NUTS & BOLTS No. 2 has 108 subprograms, 10 pp of documentation. NUTS & BOLTS #3 has 140 subprograms and 11 pp. of documentation. NOW JUST $15 EACH, POSTPAID.

TIPS FROM THE TIGERCUB

These are full disks which contain the programs and routines from the Tips from the Tigercub newsletters, in ready-to-run program format, plus text files of tips and instructions.
TIPS (Vol. 1) contains 50 original programs and files from Tips newsletters No. 1 through No. 14. TIPS VOL. 2 contains over 60 programs and files from Nos. 15 thru 24. TIPS VOL. 3 has another 62 from Nos. 25 through 32. TIPS VOL. 4 has 48 more from issues No. 33 through 41.
NOW JUST $10 EACH, POSTPAID.

```
**************************
*       NOW READY        *
* TIPS FROM TIGERCUB VOL.5 *
* Another 49 programs and  *
* files from issues No. 42 *
* through 50. Also $10 ppd *
**************************
```

TIGERCUB CARE DISKS #1,#2,#3 and #4. Full disks of text files (printer required).
No. 1 contains the Tips news letters #42 thru #45, etc. Nos. 2 and 3 have articles mostly on Extended Basic programming. No. 4 contains Tips newsletters Nos. 46-52. These were prepared for user group newsletter editors but are available to anyone else for $5 each postpaid.

This program uses the program that writes a program technique to create a program that can be used over and over to create quiz programs.

When you key in the these routines, DON'T change any line numbers!

First key in this routine and run it to create a D/V 163 file named ASCII on the disk in drive 1.

```
100 OPEN #1:"DSK1.ASCII",VAR
IABLE 163,OUTPUT
110 FOR J=1 TO 125 :: X$=X$&
CHR$(J):: X2$=X2$&CHR$(J+125
):: NEXT J
120 PRINT #1:CHR$(0)&CHR$(23
0)&"X$"&CHR$(190)&CHR$(199)&
CHR$(125)&X$&CHR$(0)
130 PRINT #1:CHR$(0)&CHR$(24
0)&"X2$"&CHR$(190)&CHR$(199)
&CHR$(125)&X2$&CHR$(130)&"J$
"&CHR$(190)&"X$"&CHR$(184)&"
X2$"&CHR$(0)
140 PRINT #1:CHR$(255)&CHR$(
255)
```

Next, key in this part -

```
220 CALL CLEAR :: CALL SCREE
N(5):: FOR SET=1 TO 12 :: CA
LL COLOR(SET,2,8):: NEXT SET
 :: DIM L$(250,4)
230 !skip to line 280!
280 READ M$ :: DISPLAY AT(2,
14-LEN(M$)/2):M$ :: FOR J=1
TO C :: READ M$ :: DISPLAY A
T(6+J,4):J;M$ :: NEXT J
290 DISPLAY AT(12,1):"Catego
ry to match? (1-"&STR$(C)&")
" :: ACCEPT AT(12,26)SIZE(1)
VALIDATE("1234"):M :: IF M>C
 THEN 290
300 IF C=2 AND M=1 THEN A=2
:: GOTO 320 ELSE IF C=2 AND
M=2 THEN A=1 :: GOTO 320
310 DISPLAY AT(14,1):"Match
against (1-"&STR$(C)&")" ::
ACCEPT AT(14,21)SIZE(1)VALID
ATE("1234"):A :: IF A>C OR A
=M THEN 310
320 DISPLAY AT(16,1):"How ma
ny choices? (2-5)" :: ACCEPT
 AT(16,25)SIZE(1)VALIDATE("2
345"):CH :: IF CH>N-1 THEN 3
20
330 FOR J=1 TO N :: FOR L=1
TO C :: READ L$(J,L):: NEXT
L :: NEXT J
340 X$=SEG$(J$,1,N):: FOR J=
1 TO CH :: RANDOMIZE :: X=IN
T(LEN(X$)*RND+1):: Y(J)=ASC(
SEG$(X$,X,1)):: X$=SEG$(X$,1
,X-1)&SEG$(X$,X+1,255):: NEX
T J
350 Z=INT(CH*RND+1):: IF L$(
Y(Z),1)=Y$ THEN 350 ELSE Y$=
L$(Y(Z),1)
360 DISPLAY AT(8,1)ERASE ALL
:L$(Y(Z),M):: FOR J=1 TO CH
:: DISPLAY AT(10+J,4):J;L$(Y
(J),A)
370 NEXT J :: DISPLAY AT(23,
1):""
380 DISPLAY AT(20,1):"(1-";S
TR$(CH);")?" :: ACCEPT AT(20
,8)SIZE(1)VALIDATE(DIGIT):Q
:: IF Q=0 OR Q>CH THEN 380
390 IF L$(Q,M)<>L$(Z,M)THEN
410 :: DISPLAY AT(23,1):"COR
RECT!"
400 CALL SOUND(100,659,5)::
CALL SOUND(100,784,5):: CALL
 SOUND(400,1047,5):: GOTO 34
0
410 DISPLAY AT(23,1):"WRONG!
" :: CALL SOUND(300,110,0,-4
,5):: GOTO 380
```

Enter MERGE DSK1.ASCII and then SAVE DSK1.QUIZ,MERGE

Then key in -

```
100 OPEN #1:"DSK1.QUIZ",VARI
ABLE 163,INPUT :: OPEN #2:"D
SK1.QUIZ/2",VARIABLE 163,OUT
PUT
110 FOR J=220 TO 410 STEP 10
 :: LINPUT #1:M$ :: CALL LIN
E(J,LN$)
120 PRINT #2:LN$&CHR$(156)&C
HR$(253)&CHR$(200)&CHR$(1)&"
1"&CHR$(181)&CHR$(199)&CHR$(
LEN(M$))&M$&CHR$(0):: NEXT J
130 PRINT #2:CHR$(255)&CHR$(
255):: CLOSE #1 :: CLOSE #2
140 SUB LINE(LN,LN$):: LN$=C
HR$(INT(LN/256))&CHR$(LN-256
*INT(LN/256)):: SUBEND
```

Run that to convert the merge file QUIZ into another merge file QUIZ/2. Then key this in -

```
100 CALL CLEAR :: CALL SCREE
```

```
N(5):: FOR SET=1 TO 12 :: CA
LL COLOR(SET,2,8):: NEXT SET
 :: DISPLAY AT(2,5):"TIGERCU
B QUIZWRITER"
110 CALL CHAR(64,"3C4299A1A1
99423C"):: DISPLAY AT(4,1):"
@ Tigercub Software for free
":"distribution - no copying
":"fee may be charged."
120 DISPLAY AT(8,1):"This pr
ogram will write":"multiple-
choice quizzes of":"the cate
gory match type."
130 DISPLAY AT(11,1):"It wil
l accept up to 250":"records
, if memory permits,":"and u
p to 4 categories per":"reco
rd."
140 DISPLAY AT(15,1):"For in
stance, a quiz on the":"tabl
e of elements could have":"t
he element name, its symbol"
:"and its atomic weight."
150 DISPLAY AT(19,1):"The pr
ogram will allow you":"to se
lect which two cate-":"gorie
s to match."
160 DISPLAY AT(23,8):"PRESS
ANY KEY" :: DISPLAY AT(23,8)
:"press any key" :: CALL KEY
(5,K,S):: IF S=0 THEN 160 EL
SE CALL CLEAR
170 DISPLAY AT(2,1):"The Qui
zwriter can be used":"over a
nd over to write any":"numbe
r of different quizzes,"
180 DISPLAY AT(5,1):"and eac
h quiz can be SAVEd":"and ru
n again and again."
190 DISPLAY AT(12,1):"Place
a disk in drive 1 with":"eno
ugh space available for":"th
e quiz."
200 DISPLAY AT(15,1):"What f
ilename will you use":"for t
he quiz?":"DSK1." :: ACCEPT
AT(17,6):F$ :: CALL CLEAR
210 OPEN #1:"DSK1."&F$,VARIA
BLE 163,OUTPUT
220 !skip to line 420!
420 DISPLAY AT(8,1):"TITLE O
F QUIZ?" :: ACCEPT AT(10,1):
T$
430 T$=CHR$(147)&CHR$(200)&C
HR$(LEN(T$))&T$ :: DISPLAY A
T(12,1):"NUMBER OF CATEGORIE
S (2-4)?"
440 ACCEPT AT(12,28)SIZE(1)V
ALIDATE("234"):C :: PRINT #1
:CHR$(0)&CHR$(250)&"C"&CHR$(
190)&CHR$(200)&CHR$(1)&STR$(
C)&CHR$(0)
450 FOR J=1 TO C :: DISPLAY
AT(12+J*2,1):"CATEGORY #";ST
R$(J);" TITLE?" :: ACCEPT AT
(13+J*2,1):C$(J)
460 T$=T$&CHR$(179)&CHR$(200
)&CHR$(LEN(C$(J)))&C$(J):: N
EXT J
470 PRINT #1:CHR$(1)&CHR$(14
)&T$&CHR$(0)
480 DISPLAY AT(2,1)ERASE ALL
:"INPUT DATA":;:"(input END
when finished)"
490 N=N+1 :: Z$="" :: DISPLA
Y AT(6,1):"RECORD #"&STR$(N)
&RPT$(" ",200):: FOR J=1 TO
C :: DISPLAY AT(7+J,1):C$(J)
:: ACCEPT AT(8+J,1)SIZE(20):
Y$
500 IF Y$="END" THEN N=N-1 :
: GOTO 530
510 Z$=Z$&CHR$(200)&CHR$(LEN
(Y$))&Y$&CHR$(179):: NEXT J
520 LN=1000+N*10 :: CALL LIN
E(LN,LN$):: PRINT #1:LN$&CHR
$(147)&SEG$(Z$,1,LEN(Z$)-1)&
CHR$(0):: GOTO 490
530 PRINT #1:CHR$(1)&CHR$(4)
&"N"&CHR$(190)&CHR$(200)&CHR
$(LEN(STR$(N)))&STR$(N)&CHR$
(0)
540 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1
550 DISPLAY AT(8,1)ERASE ALL
:"Enter NEW":;:"Enter MERGE
DSK1."&F$:;:"Enter SAVE DSK1
."&F$:;:"RUN" :: END
560 SUB LINE(LN,LN$):: LN$=C
HR$(INT(LN/256))&CHR$(LN-256
*INT(LN/256)):: SUBEND
```

Enter MERGE DSK1.QUIZ/2 and SAVE the result as your completed QUIZWRITER.

This truly remarkable one-line disk cataloger tinygram by John Martin was published in the Jackson County newsletter -

```
1 IF F THEN INPUT #1:A$,A,J,
K :: IF J THEN PRINT A$;TAB(
12);J;TAB(18);SEG$(B$,ABS(A*
2)+1,2);K;TAB(27);A<0 :: GOT
O 1 ELSE RUN ELSE B$="AVDFDV
IFIVPG" :: INPUT "DSK":F ::
OPEN #1:"DSK"&STR$(F)&".",IN
TERNAL,RELATIVE,INPUT :: GOT
O 1 ! BY JOHN M
```

And an ingenious tinygram version of Wheel of Fortune, in the West Penn newsletter.

```
1 ! *** FORTUNE OF WHEELS **
    *      A TINYGRAM      *
    * by Mike & Ed Machonis*
    ************************
2 CALL CLEAR :: INPUT "ENTER
 THE MYSTERY PHRASE    ":M$
:: CALL CLEAR :: L=LEN(M$)
3 D$=RPT$(CHR$(30),L):: FOR
J=1 TO L :: IF SEG$(M$,J,1)<
>" " THEN 4 ELSE D$=SEG$(D$,
1,J-1)&" "&SEG$(D$,J+1,L)
4 NEXT J :: PRINT D$
5 T=T+1 :: PRINT :"TRY No.";
T;:: :: INPUT "TYPE LETTER O
R ENTIRE PHRASE":A$ :: IF LE
N(A$)>1 AND LEN(A$)<L THEN 5
6 W=L+1-T :: IF A$=M$ THEN 9
7 FOR J=1 TO L :: IF SEG$(M$
,J,1)=A$ THEN D$=SEG$(D$,1,J
-1)&A$&SEG$(D$,J+1,L)ELSE 8
8 NEXT J :: PRINT :D$ :: GOT
O 5
9 FOR J=1 TO M :: CALL SOUND
(200+J*10,330+40*J,0):: NEXT
 J :: PRINT :"YOU WIN ";STR$
(W);",000 WHEELS!":;:: INP
UT "PRESS ENTER TO PLAY AGAI
N":G$ :: T=0 :: GOTO 2

100 ON WARNING NEXT :: DISPL
AY AT(3,10)ERASE ALL:"KALKUL
ATOR":"":"Input 1st value an
d Enter.":"Input other value
s preceded":"by +,-,* or / a
nd Enter." ! by Jim Peterson
101 DISPLAY AT(8,1):"Input =
 and Enter to get":"final re
sult."
110 R=14 :: C=1 :: ACCEPT AT
(12,1):N :: V=N :: F=1 :: N$
=STR$(N):: GOSUB 200
120 ACCEPT AT(12,1)VALIDATE(
"+-*/=",NUMERIC):N$ :: A=POS
("+-*/=",SEG$(N$,1,1),1):: G
OSUB 200 :: IF A=0 THEN 120
:: IF A=5 THEN 160
130 ON ERROR 140 :: N=VAL(SE
G$(N$,2,LEN(N$)-1)):: GOTO 1
50
140 CALL SOUND(200,110,5,-4,
5):: C=C-LEN(N$):: DISPLAY A
T(R,C):"" :: RETURN 120
150 IF A=1 THEN V=V+N :: GOT
O 120 ELSE IF A=2 THEN V=V-N
 :: GOTO 120 ELSE IF A=3 THE
N V=V*N :: GOTO 120 ELSE IF
A=4 THEN V=V/N :: GOTO 120
160 DISPLAY AT(R,C):STR$(V):
: F,V=0 :: GOTO 110
200 DISPLAY AT(R,C):N$ :: C=
C+LEN(N$):: IF C>20 THEN C=1
 :: R=R+1 :: RETURN ELSE RET
URN
```

Here is the world's shortest tic-tac-toe game, by R. Walters, converted to a tiny gram by Jim Peterson

```
2 DISPLAY AT(5,1)ERASE ALL:"
LET'S PLAY TIC-TAC-TOE": :"T
HE BOARD IS NUMBERED:": :TAB
(10);"1 2 3": :TAB(10);"8 9
4": :TAB(10);"7 6 5":
3 A=9 :: GOSUB 8 :: S=B
4 DEF F(X)=X-4+4*SGN(8.5-X)
5 C=F(S+1):: GOSUB 6 :: C=F(
S+3):: GOSUB 6 :: C=F(S+6)::
 IF S/2=INT(S/2)THEN 7 :: DI
SPLAY AT(20,1):"I MOVE TO";F
(S+4):"":"THE GAME IS A DRAW
" :: STOP
6 A=C :: GOSUB 8 :: H=B :: I
F H<>F(C+4)THEN 7 ELSE RETUR
N
7 DISPLAY AT(20,1):"I MOVE T
O";F(C+4);"AND WIN!" :: END
8 DISPLAY AT(20,1):"I MOVE T
O";A:"":"WHERE DO YOU MOVE T
O?" :: ACCEPT AT(22,23)VALID
ATE("12345678"):B :: RETURN

1 ! STRAIGHT-LINE CALCULATOR
    TINYGRAM by Jim Peterson
    Accepts input such as
    6+6-11*2+3/4
2 T,F=0 :: C$="+-*/" :: ACCE
PT AT(12,1)ERASE ALL VALIDAT
E(NUMERIC,C$):F$ :: L=LEN(F$
):: FOR J=1 TO L :: X$=SEG$(
F$,J,1):: P=POS(C$,X$,1):: I
F P=0 THEN 5
3 IF F=0 THEN T=VAL(SEG$(F$,
1,J-1)):: F=1 :: A=J+1 :: P2
=P :: GOTO 5
4 V=VAL(SEG$(F$,A,J-A)):: A=
J+1 :: GOSUB 7 :: P2=P
5 NEXT J :: V=VAL(SEG$(F$,A,
255)):: GOSUB 7 :: DISPLAY A
T(12,L+1):"=";STR$(T)
6 DISPLAY AT(24,1):"PRESS AN
Y KEY" :: CALL KEY(0,K,S)::
IF S=0 THEN 6 ELSE 2
7 IF P2=1 THEN T=T+V ELSE IF
 P2=2 THEN T=T-V ELSE IF P2=
3 THEN T=T*V ELSE T=T/V
8 RETURN
```

That's all, folks!

ADVENTURES INSIDE XB

Some of you have asked about uses for a GRAM device. (i.e. P-GRAM, GRAMULATOR, GRAM-KRAKER, ETC.). Primarily I believe the prime usage is that it allows you to modify cartridges to suit your needs. Those who know my penchant for modifying software (nothing is ever quite like I want) this type of device would seem a must addition to my equipment.

To introduce you to some possibilities I will note some modifications I have made to XB to make it respond more to my particular needs. For those of you who already have a GRAM device I offer a variation or two of a change most of you have already made. Along the way I may even note some of the things that get me into these things (besides my well noted laziness).

The P-GRAM manual contains a suggested change that would allow you to press the space bar after selecting XB to bypass the autoload feature. Considering the number of times I manage to have a load file in drive 1 when I wish to end up in XB made this a natural start point. To implement this change you were to change the hex string at >63CD as follows. I will display the before and after editor dump to the left and a short explanation to the right. If you entered g63CD in the editors first block the hex display in the first 12 bytes of the memory window should look like those following the address. If your display is not the same do not proceed, but use find string to locate the appropriate string.

```
>63CD 86 A3 71 06 64 8E 06 6A 84 00 06 68 ORIGINAL
>63CD 58 00 71 06 64 8E 06 6A 84 00 06 68 BR(OP-CODE 58) JUMPS TO 7800
```

Now we procede to g7800 and should find a screen full of 00. If not you are working on something other than standard TI XB.

```
>7800 00 00 00 00 00 00 00 00 00 00 00 00
>7800 86 A3 71 03 D6 75 20 63 D3 43 D0 00 SCAN(03), CEQ(D6), BS(63), BR(43)
```

This mod works fine. Now my system contains a MYARC 512 card. Over time I have converted many application disks for use with ASGARD'S RAM*BOOT program. To actuate RAM*BOOT you hold down the space bar after selecting XB. Now my poor mind was getting confused trying to determine when to press the space bar. Well why don't I modify this so any key press would bypass autoload. Maybe because I don't know anything about GPL. Now I turn to the INTERN and start learning about GPL op-codes. Somewhere I read that c8375 would contain >FF if no key press was detected. I couldn't find the reference again, but we will give it a try.

```
>7800 00 00 00 00 00 00 00 00 00 00 00 00
>7800 86 A3 71 03 D6 75 20 63 D3 43 D0 00 SCAN(03), CEQ(D6), BS(63), BR(43)
>7800 86 A3 71 03 D6 75 FF 63 D0 43 D3 00 SCAN(03), CEQ(D6), BS(63), BR(43)
```

Use the bottom >7800 entry if you prefer this method and haven't allready entered the space bar change. I included both for reference. Now all I do is hit any key after the XB selection, or I hold the 2 down till ready appears (I'm lazy).

The next suggestion in the manual talks about changing the autoload file name. My 512 stays alive on an external supply so it seemed natural to boot from the ram disk. The preferable load file would be DSK5.LOAD1 for my purposes. The RAM*BOOT disks I make have the program load file changed to LOAD1 because it fits nicely between RAM*BOOTs LOAD and LOAD2 files and is easier for me to remember if I just want the application program to load from another drive. The manual says that you can change the load file name if you maintain a 4 letter name. Not really wishing to redo RAM*BOOT on multiple disks I set out to find another way. I discovered a spare byte at the end of LOAD so I

put a 1 in it. Now I incremented the size byte by 1 and went off to try it. SYNTAX ERROR just what I always wanted. Not to worry that just meant something else needed to change. I set the search string to find a spot that called for the load name size byte. A little dumb luck didn't hurt and I found one place that called for that address. Looking around I discovered what looked like another size byte but >2 greater then the load name, so I added 1 to it. Now the strings look like this.

```
>6351 09 44 53 4B 31 2E 4C 4F 41 44 00 00 *DSK1.LOAD**
>6351 0A 44 53 4B 35 2E 4C 4F 41 44 31 00 *DSK5.LOAD1*

>648E 31 00 0B A8 20 63 51 BF 2C 08 20 44
>648E 31 00 0C A8 20 63 51 BF 2C 08 20 44 File length + >2
```

This worked just fine. When I let XB autoload whatever was resident on the ram disk appeared. In fact if you wish you could even extend the load name further if you wished by keeping the string size indicators >2 appart. You will however note a slight change in opperation. Your cursor will change shape. The cursor definition is the 8 bytes starting at >635C. followed by the 8 byte edge character starting at >6364. Give you ideas about a customized cursor? Try replacing the edge character with FF 81 81 81 81 81 81 FF and watch the boxes move up the sides of the screen as you type in a program. This will also tell you how wide your horizontal overscan is. Just an added attraction along the way.

Everything now works and all is well. NOT AT MY HOUSE! After spending the latter part of an evening troubleshooting a friends XB cartridge (see a future article on using a GRAM device for this purpose) I left an XB cartridge in the WIDGET but the switch in an unused position. Next day my office phone rings, my kids decide to use the computer, DAD the computers broke! Whats wrong? When I start XB I get a funny screen (funnlweb). No problem just reset, hold the 2 key down until the READY appears and the enter RUN"A.LOAD". About the same call I got when I put the MG prom set in my COR COMP controller.

When I got home they were still unhappy about losing the autoload they were accustomed to. I thought the young ones were supposed to be more tolerant of changes! Well there were times that I became agitated at needing to go to XB to load from drive 1 so I changed things back and set off to have the best of both worlds.

After much more time buried in the INTERN to increase my meager knowledge of GPL and discovering how lucky I had been to locate the correct place to change to make the longer file name work. It seems I found one of the few GPL op-codes (MOVE or 31) that requires a full word address. Well I managed to learn enough to make the following mods. First go to >648E and check the string to make sure there is a >BF at >6495.

```
>648E 31 00 0B A8 20 63 51 BF 2C 08 20 44
```

Now go to g63CD and make the following mod. I am assuming you already put the autoload bypass change in. If not make the change to 58 00 noted above.

```
>63CD 58 00 71 06 64 8E 06 6A 84 00 06 68
>63CD 58 00 71 06 64 95 06 6A 84 00 06 68, CALL(06) at 6495
```

This change calls the autoload subroutine after the load file info has been moved. In other words it bypasses moving the DSK1.LOAD name and >09 size information at this point.

Now we move on to g7800 which should be all 00 except for the prior modification.

```
>7800 86 A3 71 03 D6 75 FF 78 54 D6 75 35 **q**u*xT*u5
>780C 78 B4 D6 75 42 78 CC D6 75 62 78 CC x**uBx**ubx*
>7818 43 D3 00 00 00 00 00 00 00 00 00 00 C***********
>7824 00 00 00 00 00 00 00 00 00 00 00 00 ************
>7830 00 00 00 00 00 00 00 00 00 00 00 00 ************
>783C 00 00 00 00 00 00 00 00 00 00 00 00 ************
>7848 00 00 00 00 00 00 00 00 00 00 00 00 ************
>7854 31 00 0B A8 20 63 51 05 63 D0 00 00 1*** cQ*c***
>7860 00 44 53 4B 31 2E 00 00 00 00 00 00 *DSK1.******
>786C 31 00 00 A8 20 78 78 05 63 D0 00 00 1*** xx*c***
>7878 00 44 53 4B 32 2E 00 00 00 00 00 00 *DSK2.******
>7884 31 00 00 A8 20 78 90 05 63 D0 00 00 1*** x**c***
>7890 00 44 53 4B 33 2E 00 00 00 00 00 00 *DSK3.******
>789C 31 00 00 A8 20 78 A8 05 63 D0 00 00 1*** x**c***
>78A8 00 44 53 4B 34 2E 00 00 00 00 00 00 *DSK4.******
>78B4 31 00 0C A8 20 78 C0 05 63 D0 00 00 1*** x**c***
>78C0 0A 44 53 4B 35 2E 4C 4F 41 44 31 00 *DSK5.LOAD1*
>78CC 31 00 0D A8 20 78 D8 05 63 D0 00 00 1*** x**c***
>78D8 0B 44 53 4B 35 2E 4C 4F 41 44 2F 42 *DSK5.LOAD/B
```

This sequence was designed with expansion and modification in mind, so rather than attempt to put the op code info to the right I left the ASCII presentation so you can see the displayable characters. If you look closely I left room for approximately 10 key press comparisons. I started the sequence for movement of file names and the names on seperate lines to make the memory location easy to find while I was in the editor. To start I assumed that I might want to autoload from any of my 4 physical drives, then I added 2 load files that are normally on the ram disk.

We will walk through some of the sequences so you can modify this to suit your system. I only left room for 6 letters in the file name you may want more. To start we will always have 86 A3 71 which clears the autoload needed flag at v371, next is a keySCAN 03, then we Compare EQual D6 at the memory location holding the key code c8375 75 with no key pressed FF, then if the condition bit has been set (the comparison was equal) we Branch on Set 78 to g7854 54. Following along the line you will note repetitive sequences, D6 75 XX 78 XX, this is accomplishing a comparison with various key codes that may be held at c8375. In the case above I want to know if I pushed a 5, B or b. If one matches the condition bit will be set and then BS (78) will move to the appropriate load file movement sequence for the filename desired. If none has matched the condition bit will not have been set so we branch on reset 43 to g63D3 D3 and bypass autoload. Remember the first comparison looked to see that a key was pressed (or no keypressed).

To accomplish the transfer of the desired file names we will examine the sequence at g78B4, first we MOVE 31 the desired number of bytes 00 0C (remember this is always 2 greater than the size of the complete file name), to the destination A8 20, from the source 78 C0, then we jump or Branch (B) 05, to the point that calls the autoload routine 63 D0. Now if we move to g78C0 and look at the string that we moved to A820. First is the size byte 0A which is the length of the string in hex, then the string itself 44 53 4B 35 2E 4C 4F 41 44 31 00 or in ASCII, DSK5.LOAD1*. I do not know why the MOVE routine takes one more byte than seems necessary, but without it you end up with a SYNTAX ERROR. It seems to work fine even if you move more bytes than required, I assume but have not worked through the next routine (autoload) to insure that it only reacts to the string size byte for the file name size. So if you are squemish about that kind of thing than leave more space between the sequences.

To customize this for whatever autoload file names you wish to use only requires that

you determine what key press is appropriate, convert the ASCII code to hex and insert in the CEQ string D6 75 XX, add the location where you are putting the MOVE sequence 78 XX, then fill in the size to MOVE 31 00 XX and the location of the string size byte A8 20 XX XX and the appropriate size and file name.

Prior to ending this session we will discuss how the BS and BR instructions work so you can accomplish this change in another GRAM if you desire. You may have acquired a version of XB that has had additional changes and you discover that g7800 has something other than 00 filling the space that I have shown you.

First you need to find sufficient empty space in a GRAM, g6000 - gDFFF, I found one at gB600. Once you have determined the location you need to change the spot that contains 86 A3 71 and change it to BRANCH (B) to the desired location 05 XX XX or your GRAM location, mine was 05 B6 00. At this point you can enter just what we have before with slight modification.

The BS instruction will change. It seemed routine at 78 XX but is not quite that easy. The BS op-codes are >60 - >7F and the first nibble (6 or 7) defines the start of an adder to the base GROM address. The key is in the last bit of the nibble (xxxa) where, a, will be a 1 or 0. This number plus the next nibble and byte define the adder. In the examples above this equates to 78 XX is saying 6000 plus 18XX. Remember that each GRAM is a seperate entity so the BS instruction works within each GRAM memory block. The start point for the 4 XB GRAMs is g6000, g8000, gA000 and gC000. Once you have determined what address you are jumping to for the routine you can select the appropriate BS op-code. In my case I jumped to B600 so I used 76 XX (A000 + 16XX). The last change you will make will be the point where the above example branches on reset (BR) 43 D3. This can only occur in the GRAM you are working in so we must Branch (B) op-code 05 to the required address 63 D3. Just substitute 05 63 D3 for the 43 D3 at the end of the key comparison sequences.

I hope I gave you sufficient information so you can decide what you would like and modify it to suit your uses. The next thing I will add is D6 75 20 78 54 so I can select XB and hold down the space bar to let DSK1.LOAD run and activate RAM*BOOT which is what started this trek to begin with.

Be sure and save as large files when you are finished so you will have a backup to load when you need.

If you have MECHATRONIC's XBII+ loaded, the same changes should work for it. In fact assuming that XBII+ is built using PROMs and GROM simmulation rather than real GROMs you and a friend who is handy with a PROM blower and code insertion could make the changes in a new PROM and wouldn't require a GRAM device. Yes such things are possible. Did anyone catch the device name I told my kids to use (RUN"A.LOAD")? Yes, my physical drive 1 will respond to that, in fact it thinks it is DSK1, dsk1, A, a and 1. Drives 2-4 think the same thing for their respective letters or numbers. Thanks to Gary Bishop's earlier article and his modification of my MG prom set.

For those who may be considering a GRAM device or who are leery of tinkering with some software that doesn't quite work the way you would like. Remember that this was accomplished by a sole who is not experienced in GPL and is only starting to work in assembly. I stumbled along the way, accomplished the objective and learned a lot in the process. So take a well backed up file, the right manuals and dive in.

Try what you wish of these ideas, if you have trouble ask me and I'll give you a hand. Enjoy your computer and have it work like you would like, it can be done.

Jerry Canady

NEW PRODUCTS FOR OUR T.I. 99/4A and 9640.....

THE ADVENTURE REFERENCE GUIDE by Mickey Schmitt

The Adventure Reference Guide is the "bible" of TI adventure gaming. This remarkable reference is the product of nearly a year of planning, research and writing. This is a large book that lists the almost 200 adventures available for the 99/4A, rates them within their categories, lists the equipment needed to run them, reviews the most significant programs, and even provides lists where they can be obtained. Did you know that most 99/4A adventures can be had for next to nothing? This book will show you where to get them and how. 108 pages. $14.95 plus $2.00 S/H write: ASGARD, P.O. Box 10306, Rockville, MD 20850 Phone: (703) 255-3085

The TI ADVENTURE COMPENDIUM Disk Set is a set of Fairware and public domain adventures (some requiring Game Modules) available on 10-SSSD disks for $19.95, 5-DSSD disks for $15.95 or 3-DSDD disks for $11.95. These are available from either:

WESTERN NEW YORK 99ers
% Harry T. Brashear
2753 Main Street
Newfane, NY 14108

M.U.N.C.H.
% Jack Sughrue
Box 459
East Douglas, MA 01516

I counted 79 adventures listed in this compendium.

GIANT ARTIST POSTERS (G.A.P.) by NAMELOC SOFTWARE

Paul Coleman has a new program written in "C" that is a support program for T.I. -ARTIST. You can take a full T.I.-ARTIST screen and print it out as a poster in any size from 10"x14" to over 5'x8' ('=feet). You can even print two full screens side by side for an even bigger poster. Load or create a picture using T.I.-ARTIST that takes up a full screen. Then save it as an instance . G.A.P. will load and print it out in banner fashion in a wide choice of sizes. Requires T.I.-ARTIST V2.0, disk,32K, and an EPSON compatible printer. (and lots of ribbons). Cost $10. plus $1.50 S/H and available from NAMELOC SOFTWARE, 3971 S.E. Lincoln, Portland, OR 97214

THE GEOMETER'S APPRENTICE by McCann Software

Mike McCann has produced a 3-D C.A.D. color, light and magic show for the T.I. 99/4A and GENEVE (9640). You can create screens of Lambert shaded objects. The 99/4A can be used to create motion sequences in bit-map mode. The 9640 will be strained to it's limits in all available memory use and color use in 512x212 mode. Objects may be scaled, translated and rotated in 3-D and once created moved from one drawing to another. Both the 99/4A and the 9640 versions are menu driven with the added bonus of the 3-D slides language on the 4A. T.G.A. produces external files compatible with The Printers Apprentice and will print using "Cpixel" user defined pixel shapes in portrait or landscape modes on popular dot matrix printers. Required for T.I. 99/4A: 32K mem., Disk,and XBasic/Ed.Assembler. Required for Geneve (9640): at least MDOS version 1.01, V.99 GPL and E/A. Printers: 100% Epson, Gemini 10X, Star NX, IBM, Panasonic 1091, and T.I. Send check or M.O. for $39.95 to: McCANN SOFTWARE, P.O. Box 34160, Omaha, NE 68134

PAGE PRO 99 by ASGARD SOFTWARE

A remarkable program that lets you compose a 66 line page full of text, graphics and lines quickly and easily. It's a "what-you-see-is-what-you-get" program with the ability to paste in pictures and type text. Up to 28 pictures of any size or shape anywhere on the page. Type text in any number of large and small and upper or lower case fonts and draw lines anywhere needed.You can type up,down,left or right with full editing. You can window around the page using TI-WRITER keys, and load or save as text file. You can even load a text file and paste it on the page. There are many more options and features. For TI 99/4A and Geneve. $24.95 plus $.75 S/H. ASGARD SOFTWARE, P.O. Box 10306, Rockville, MD 20850

NEXT MEETING

MONDAY, AUGUST 14

6:30 PM --- WEST MUSIC COMPANY

NEW FORMAT -- SHORT BUSINESS

MEETING, LOTS OF DEMONSTRATIONS!

COME LEARN SOMETHING NEW!

Cedar Valley 99'er Users Group
377 Cambridge Dr. NE
Cedar Rapids, Iowa 52402

PLEASE NOTE NEW ADDRESS!
EFFECTIVE AUGUST 1st

Send To:

GARY BISHOP
124-222
3270 28TH AVE
MARION IA 52302