ROCKY MOUNTAIN 99'ers

TIC TALK

VOL III, NO 6 DENVER, COLORADO USA FEB 1985
Non-member Subscription Rate - $7.50 Annually Single Copy Price - 75 cents

FROM THE EDITOR

I want to apologize to all for last month's newsletter. A new printer was tried, and we all know why their price was so much better. Also, I apologize for the omission of a portion of the TI-Writer article. The article had a transliterate included, and I forgot that the printer would just consider it as such and not print it out. The transliterate is: .TL 126:27,75,6,0,16,32,126,32,16,0 .

At the meeting this month, Mike Holmes will discuss his program Q-TERM and demonstrate it.

Does anyone out there know the whereabouts of the club's Disk Manager cartridge? It sure would be helpful with the newsletter, since I don't have my own disk system yet.

For the benefit of those not in the habit of looking at the return address on their newsletter,we have a new one. At least for any correspondence to the editor. The old address is still good for correspondence to the club members, etc.

I will be in the process of moving in the next couple of weeks, and you may find it difficult to get me on the telephone. I apologize in advance for this, but it is necessary for my job. Anyone that needs to reach me can call 477-6034 and ask for Gloria. She can get a message to me no matter what.

See you on the fifth!

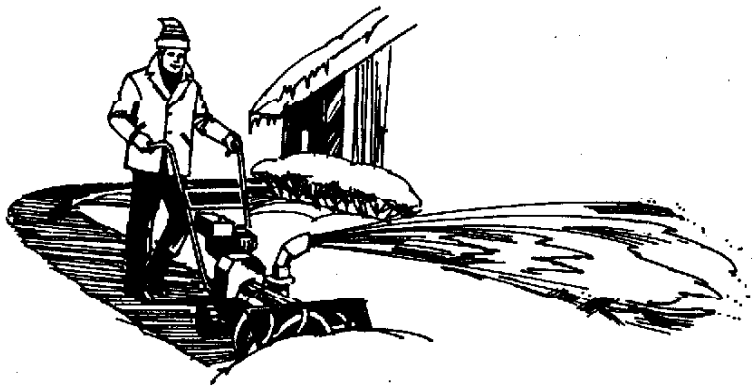
FEBRUARY MEETING

FEBRUARY 5

Jefferson County Fairgrounds

Auditorium 7:00 PM

6th Ave. West to Indiana Ave.



GRAPHICS! From Northeast Iowa UG
By Barb Berg

When I first saw the TI, the one thing about it that impressed me most was the ease with which graphics could be displayed on the screen. And, being somewhat of an artist, I immediately dove into all the graphics capabilities. With this in mind, it may surprise you that this month's column was a little difficult to put together. If you have played with displaying any graphics character, and possibly made it move across the screen, then you probably already know most of the basics in this area. What I wanted to do in this column was show you something a little different, hopefully something that you hadn't already tried.

Once a character is on screen, moving it is simply a matter of advancing it one square in the direction you want it to. For an example, examine the "Inchworm" program or "Bouncing Ball" in the URG. Both show a method of moving a character on screen. In "Inchworm", the character that appears to move is replaced by a blank space to avoid drawing a line across the screen. Many programs have been written in Basic that use this replacement method, either with the space or by using another character, as to show the path that the character moved.

One thing that I haven't seen much used, however, is the effect that can be accomplished by changing a character already on screen. Here is an example of one such effect that can be used. I call it "Dark Eyes".

```
100 CALL CLEAR
110 X=112
120 FOR I=0 TO 8
130 READ A$(I)
140 IF I=8 THEN 180
150 CALL CHAR(X,A$(I))
160 X=X+1
170 NEXT I
180 CALL SCREEN(2)
190 CALL COLOR(11,1,1)
200 PRINT TAB(10);"pqr   pqr":::::TAB(12);"stuvw"::::::::
210 CALL COLOR(11,10,1)
220 FOR I=0 TO 14 STEP 2
230 GOSUB 310
240 NEXT I
250 FOR I=16 TO 0 STEP -2
260 GOSUB 310
270 NEXT I
280 FOR DL=1 TO 500
290 NEXT DL
300 GOTO 220
310 J=0
320 FOR X=112 TO 114
330 CALL CHAR(X,SEG$(A$(8),1,I+2)&SEG$(A$(J),I+3,LEN(A$(J))))
340 J=J+1
350 NEXT X
360 FOR X=115 TO 119
370 CALL CHAR(X,SEG$(A$(J),1,16-I))
380 J=J+1
```

```
390 NEXT X
400 RETURN
410 DATA 01030F1F3F7F0F03,FFC3810000000081,80C0E0F3F8FCF0C0,0007070301000000
,00F7F7F7F7F77030
420 DATA 00EFEFEFEFEF0000,00DFDFDFDFDE1C18,00C0C0800000000,0000000000000000
```

Lines 110 to 170 contain the loop that reads the character data contained in the last two lines of the program and defines the characters in set 11. A$(8), however, is not used to define a character yet, but will be used instead as a string to take segments of later. Lines 180 and 190 produce a black screen and make the characters we are using transparent until they are printed on the screen in line 200. Since we are using small letters as our redefined characters, we can display them faster this way than by using CALL HCHAR. There are three spaces between the two "pqr" characters. These will be the eyes, and "stuvw" will be a leering smile. Line 210 "turns on" the characters, making them visible. 220 through 270 are loops which branch to a subroutine where all the characters are redefined. 280 and 290 are a delay loop, followed by a return to line 220 where the whole thing starts all over again. To stop the program, therefore, it is necessary to press FCTN 4 (CLEAR).

Lines 310 to 400 are the heart of the program, and contain the subroutine for redefining. J is set equal to 0 (zero), the first member of our subscripted string array with the character definitions. 320 begins the loop for the eye characters. Line 320 redefines the eyes by combining the first I+2 characters of our blank string, A$(8), which consists of 16 zeros, with the ending segment of each character's string that begins at I+3. The first time through the loop at 220, I+2=2, so we will make the first 2 characters of each eye definition zeros. Then we make the rest of the character the same as it was by keeping the part of that definition that begins at I+3, or 3 in this case, and ends with the last character in the defining string. Since the loops in 220 and 250 increment and decrement by two, we are only changing one bit-row at a time.

Line 370 does something similar to the smile characters, except that here we take the first 16-I characters of each definition. The remaining zeros are assumed, so none are attached to the end. The effect is of slowly closing and opening eyes and mouth of a face in the dark. This method is somewhat slow for so many characters, but you can see that a different type of effect can be made in this manner. You could make a character seem to sink into a pool, for example, by combining zeros and the first 16-I characters of your definition for that character. The loop in line 220 would be changed to FOR I=2 to 16 and line 330 would read

```
CALL CHAR(X,SEG$(A$(8),1,I)&SEG$(A$(J),1,16-I)))
```

in order to achieve this effect. When I=2, the first two characters of the definition are zero and the last 14 are the original first 14.

Now, something a little different for those with EXTENDED BASIC and sprites. Try the following:

```
100 REM * BUBBLES *
110 CALL CLEAR :: CALL CHAR(100,"3C42995A5A99423C"&RPT$("0",48),108,"994200
8181004299"&RPT$("0",48))
120 CALL CHAR(104,"0718204040808080808080404020180FE01804723A1909010101010202
20418E0")
```

```
  130 CALL  CHAR(112,"8141201000000 1C3C3010000102041818182040800000 80C3C380000
8048281")
  140 CALL SCREEN(15):: CALL MAGNIFY(3)
  150 CALL SPRITE(#1,100,6,165,80,#2,104,16,135,160)
  160  FOR X=9 TO .01 STEP -.08 :: Y=COS(X):: X1=X*15 :: Y=Y*65 :: Y=ABS(100-Y)
:: CALL LOCATE(#1,X1+30,Y/2,#2,X1,Y):: NEXT X
  170 CALL POSITION(#1,R1,C1,#2,R,C):: CALL DELSPRITE(#1,#2):: CALL  SPRITE(#3,
108,16,R1,C1,#4,112,16,R,C):: CALL DELSPRITE(#3,#4)
  180 FOR DEL=1 TO 100 :: NEXT DEL :: GOTO 150
```

Lines 110 to 130 define the sprites. Character 100 is a small bubble, and 108 is its "pop". 104 is a large bubble, and 112 is its "pop". The screen is set to gray and the sprites are double sized, unmagnified, as the large bubble is defined by four characters. If we had used a magnification of 4, the bubble would have been too large to look realistic. A smaller magnification would only show the top left corner of the large sprites. Line 150 defines sprites 1 and 2 as the two bubble characters and places them on the screen.

Line 160 is the meat of this program. The loop counts down from 9 to .01 to achieve the effect of the bubble rising from the bottom to the top of the screen. Y is set equal to the cosine of X. What this does is this: the bubble will follow a path similar to a sine wave pattern. Y=SIN(X) achieves similar results. Since the cosine of our numbers would produce too small a sine wave pattern, we enlarge the pattern by making X1=X*15 and Y=Y*65. Both X1 and Y could be multiplied by the same number, such as 30, but I liked the effect produced by the factors used here. (If you want to see what changing these numbers does to the wave pattern, change them.)

Y is then set equal to the absolute, or positive, of 100-Y. ABS is used case we should happen to get a negative number, or if you should happen to use a formula other than SIN or COS to see what results you get. (Some possibilities are on page 202, Appendix K, of the EXTENDED BASIC manual.) 100-Y keeps the bubble in the range of screen I happened to use. Another value could be used to achieve a different effect.

CALL LOCATE puts the sprites at the locations figured by the formulas. #1 rises slightly differently than #2 by the formulas contained in this statement. The loop repeats until the sprite is at the top of the screen.

CALL POSITION returns the dot-row and dot-column of the upper left-hand corner of each sprite in variables R1, C1, R and C, so we have the positions we will need to locate the "pops" for each bubble. The bubbles are removed, their "pops" are positioned and then immediately removed for the effect desired. Again, a delay loop holds the screen blank temporarily before the whole thing starts all over again.

If the loop in 160 were FOR X=.01 to 9 STEP .08 instead, the bubble would fall from the top of the screen. This would make an interesting effect for a falling leaf or sheet of paper. After you have typed in the above program and watched it run, make one change in one of the places I suggested, and notice the difference. If you see one you like, keep that routine for another program. Play around with it a little and have fun.

My next column will contain more information on sprites, including some of the more mysterious commands which may have left you in the dark. In the meantime, you have something you can do some experimenting with. Try some of the challenges below, and until next time, happy computing!

SHOOT THE MAN DOWN
BY STEPHEN JOHNSON
CHANNEL 99

The principal of most arcade games is simply to shoot something. Sounds very simple, but there are many hidden pitfalls.

Your bullet should leave the gun not jump off the ground beside it. The bullet should destroy the target if it hits it.

It's all very easy. BUT how is it done? In basic the greatest problem is speed, actually firing a bullet that walks across or up the screen has to be "PRINT" and "DELETE", it is inherently slow.

Subroutines should be short precise and use no "GOSUBS".

Below you will see two methods in Basic, one using the "CALL VCHAR" for the laser shot. This method is the fastest.

In Extended Basic there are literally dozens of methods, even registering the hit has four or five methods.

Because of the limitation of space I will only show one method here.

All of the following routines could be installed into your programs, or you could write a program around one or more of them.

```
100 REM SHOOTING EXAMPLE
110 REM FOR CHANNEL 99 USERS GROUP
120 REM T.I. BASIC
130 REM BY STEPHEN JOHNSON
140 CALL CLEAR
150 CALL CHAR(129,"18183C3C7EFFE766")
160 CALL CHAR(136,"00000000187EFF66")
170 CALL CHAR(137,"1010101010101010")
180 DATA 16,16,16,16
190 READ MYCL,MYOCL,UCL,UOCL
200 CALL JOYST(1,X,Y)
210 CALL KEY(1,K,S)
220 MYCL=MYCL+SGN(X)
230 IF (MYCL>0)*(MYCL<33)THEN 250
240 MYCL=MYOCL
250 UCL=UCL+INT(RND*3-1)
260 IF (UCL>0)*(UCL<33)THEN 280
270 UCL=UOCL
280 CALL HCHAR(1,UOCL,32)
290 CALL HCHAR(1,UCL,136)
300 CALL HCHAR(24,MYOCL,32)
310 CALL HCHAR(24,MYCL,129)
320 MYOCL=MYCL
330 UOCL=UCL
340 IF K<>18 THEN 200
350 CALL HCHAR(23,MYCL,137)
360 FOR R=22 TO 1 STEP -1
370 CALL HCHAR(R+1,MYCL,32)
380 CALL HCHAR(R,MYCL,137)
390 NEXT R
400 IF MYCL=UCL THEN 430
410 CALL HCHAR(1,MYCL,32)
420 GOTO 200
430 PRINT "YOU GOT HIM"
440 END
```

```
100 REM SHOOTING EXAMPLE
110 REM FOR CHANNEL 99 USERS GROUP
120 REM T.I. EXTENDED BASIC
130 REM BY STEPHEN JOHNSON
140 CALL CLEAR
150 CALL CHAR(129,"18183C3C7EFFE766")
160 CALL CHAR(136,"00000000187EFF66")
170 CALL CHAR(137,"1010101010101010")
180 CALL SPRITE(#1,129,2,180,128,#2,136,
15,1,1,0,5,#3,137,2, 200,1)
190 CALL JOYST(1,X,Y)
200 CALL KEY(1,K,S)
210 CALL MOTION(#1,0,X)
220 IF K<>18 THEN 190
230 CALL POSITION(#1,X,Y)::
CALL LOCATE(#3,X,Y):: CALL MOTION(#3,-12,0)
240 FOR L=1 TO 50
250 CALL JOYST(1,X,Y):: CALL MOTION(#1,0,X)
260 CALL COINC(#2,#3,10,C):: IF C THEN 300
270 NEXT L
280 CAL MOTION(#3,0,0)
290 GOTO 190
300 PRINT"YOU GOT HIM"
310 END
```

** COINCIDENCE BY IAN JOHNSON **

```
       DEF  COINC
       REF  VSBW,VSBR,VMBW,GPLLNK
       REF  KSCAN,SOUND,VWTR
STATUS EQU  >837C
FAC    EQU  >834A
NUM    EQU  >837A
MYRND  EQU  >83D6
VDPSTA EQU  >837B
ZERO   DATA 0,0,0,0
SPRDES DATA >0400
SPRIT1 DATA >0288,>8001
SPRIT2 DATA >2810,>840F,>D000
CHARS  DATA >187E,>FFFF,>FFFF,>FFFF
       DATA >FF7E,>3C18,>1800,>1818
       DATA 0,0,0,0,0,0,0,0
       DATA >2010,>FFFF,>1020,>0000
       DATA 0,0,0,0
       DATA >0402.>FFFF.>0204.>0000
       DATA 0,0,0,0
VTAB1  DATA >0780
VTAB2  DATA >0785
HITMSK DATA >2000
EXPSND DATA >9A90
FIRBUT BYTE 18
XVEL2  BYTE 125
RED    BYTE 6
GREEN  BYTE 3
XMAX   BYTE >F8
OFFSND BYTE >FF
       EVEN
COINC  LIMI 0
       LI   R0,>E201
       MOVB R0,@>83D4
       SWPB R0
       BLWP @VWTR
       MOV  @SPRDES,R0
       LI   R1,CHARS
       LI   R2,64
       BLWP @VMBW
       LI   R0,>0300
       LI   R1,SPRIT1
       LI   R2,10
       BLWP @VMBW
       LI   R2,>0200
       MOVB R2,@NUM
       MOV  @VTAB1,R0
       MOVB @MYRND,R1
       LI   R1,>D000
       BLWP @VSBW
       LIMI 2
       LI   R1,30000
DELAY3 DEC  R1
       JNE  DELAY3
CALKEY LI   R1,>0100
       LIMI 2
       LIMI 0
       MOV  R1,@>8374
       BLWP @KSCAN
       CB   @>8375,@FIRBUT
       JEQ  FIR
       JMP  CALKEY
FIR    MOV  @VTAB2,R0
       MOVB @XVEL2,R1
       BLWP @VSBW
       LIMI 2
FFHIT  MOVB @VDPSTA,R2
       COC  @HITMSK,R2
       JNE  MISSED
       LIMI 0
       MOV  @VTAB1,R0
       LI   R1,ZERO
       LI   R2,8
       BLWP @VMBW
       LI   R3,500
       LI   R0,>0384
BANG   MOVB @RED,R1
       BLWP @VSBW
       MOVB @EXPSND,@SOUND
       MOVB @EXPSND+1,@SOUND
       MOV  R3,R2
DELAY1 DEC  R2
       JNE  DELAY1
       MOVB @GREEN,R1
       BLWP @VSBW
       MOV  R3,R2
       MOVB @OFFSND,@SOUND
DELAY2 DEC  R2
       JNE  DELAY2
       DEC  R3
       JNE  BANG
       B    @COINC
MISSED LI   R0,>0305
       LIMI 0
       BLWP @VSBR
       LIMI 2
       CB   R1,@XMAX
       JL   IFHIT
       MOV  @VTAB2,R0
       CLR  R1
       LIMI 0
       BLWP @VSBW
       LI   R0,>0304
       LI   R1,SPRIT2
       LI   R2,6
       BLWP @VMBW
       B    @CALKEY
       END
```

THE TI 99/4A PERIPHERAL EXPANSION BOX DISSASSEMBLY PROCEEDURE.

by John B. Colson who assumes no responsibility if anyone gets in trouble with this information.

Do you need to get inside your PEB? Some have wanted to increase the power output of the power supply. One reason for more power is to allow using half height disc drives that require more than the 12 volt 500 milliampers available from the TI supply. Maybe you have some foreign material stuck inside. Maybe you need to do maintenance on this unit. Whatever the reason, to many observers the disassembly appears mysterious. The clue is that the sides and front are removed as a single unit.

If you are going to upgrade the power suply output, you need to have obtained in advance the necessary components. These components depend upon how much power you are going to need. The voltage regulator that is in the TI unit can give one amper if properly heat sunk. For one and one half ampers, a Motorola MC7812C can be used. For three ampers, a Fairchild ua79H12KC can be used. For five ampers, a $20.00 RCA SK9341/933 should work. To provide a good heat sink I used the fan side of the power supply housing.

To start this proceedure: shut off the power and allow the capacitors at least a full minute to discharge internally instead of into you. Remove the power cord. Remove the disk drive(s). Take off the top cover by pushing the two back clips forward and tilting cover forward. If you have external cables attached to peripheral cards they are to be disconnected, i.e., printer, external disk drive, etc. Disconnect the internal disk drive ribbon cable and remove all cards from the box. The power supply is on the left side of the box as viewed from the front. This is obvious, but as we turn the box over, we need to keep track of where the power supply is located. Remove six

screws from the bottom of outside flanges and one screw from the bottom of the power supply housing. Remove one screw from each end of the box. Place the box on its face on a protective, soft, (non-scratching) surface and remove three screws from back of disk housing and three screws from back of power supply housing. Now the two pieces are separated and you should be able to lift the back and bottom with the heavy power supply straight up and away from the face and sides.

All the above voltage regulators come in TO3 packages and will mount at 45 degrees in the lower outside corner of the fan side. To preform this drilling, the power supply circuit board should be removed. This is accomplished by loosening two phillips screws at the inside base of the plastic right angle bracket holding the circuit board and sliding the bracket with the board to the outside. By snapping the ears on each of three plastic connectors away from each other, the connectors can be tilted toward the foil side of the board and removed. Now the remaining cables can be put out of the way under the fan and the drilling can proceed. To play safe from case ground, I electricaly insulated the voltage regulator and used silicon pad and heat sink compound on my mount. Carefully! remove the voltage regulator (the only TO3 package) from the TI circuit board and attach three wires that will reach your newly mounted voltage regulator. Note well where each wire belongs and attach them to externally mounted voltage regulator (with pins inside). Now you may re-assemble and see if it works. Mine did.

1 January 1985 Disk 1910

These Tips are distributed to Users' Groups in exchange for their newsletters - and in the faint hope that someday, somewhere, someone may buy some of my original programs. I have over 130 of them, at only $3 each - some of the users'groups charge their own members almost that much for public domain programs! My catalog costs a dollar, refundable on your first order, or refundable anyway if you ask. I give one-day service by 1st Cl. mail, I give bonus programs for repeat orders, I give free programs on disk orders, and I'm still not getting any orders!

I'm told that someone actually found a practical use for my number-scrambling rutine, so here is an expanded version. It will scramble any sequence beginning with 1 and ending with any number less than 256 or any number greater than 256 which is evenly divisible by any number less than 256 and greater than 1, within the limits of computer memory. In Extended Basic with Memory Expansion, the limit is about 10,700; if you reformat it to Basic and run it bare bones, you might get close to 13,000.

```
100 CALL CLEAR :: OPEN #1:"P
IO",OUTPUT
110 INPUT "HIGHEST NUMBER? "
:HN :: IF HN<256 THEN TN=HN
:: XX=1 :: GOTO 150
120 FOR TN=255 TO 2 STEP -1
:: IF HN/TN=INT(HN/TN)THEN 1
40
130 NEXT TN :: PRINT HN;"IS
NOT DIVISIBLE BY":"ANYTHING
LESS THAN 256  - ":"CANNOT U
SE" :: GOTO 110
140 XX=HN/TN
150 DIM M$(50)
160 CALL CLEAR :: FOR J=1 TO
 TN :: M$(1)=M$(1)&CHR$(J)::
 NEXT J :: FOR J=1 TO XX ::
M$(J)=M$(1):: NEXT J :: FOR
J=1 TO HN :: TT=1+INT((J-1)/
255)
170 RANDOMIZE :: X=INT(XX*RN
D+1):: IF LEN(M$(X))-0 THEN
170
180 Y=INT(LEN(M$(X))*RND+1)
190 PRINT #1:ASC(SEG$(M$(X),
Y,1))+TN*(X-1);
200 M$(X)=SEG$(M$(X),1,Y-1)&
SEG$(M$(X),Y+1,LEN(M$(X))):::
 NEXT J
```

Here's a little routine you can use to jazz up your title screen or text.

```
100 CALL CLEAR
110 DATA "THIS IS A DEMONSTR
ATION","OF THE","TIGERCUB SO
FTWARE","TWO-WAY PRINT ROUTI
NE"
112 FOR T=1 TO 4
113 READ M$
120 IF LEN(M$)/2=INT(LEN(M$)
/2)THEN 135
130 M$=M$&" "
131 GOTO 140
135 M$=M$&"  "
140 L=LEN(M$)
150 C=16-L/2
160 FOR J=L/2 TO 1 STEP -1
170 CALL HCHAR(10+T*2,C+J,AS
C(SEG$(M$,J,1)))
180 CALL HCHAR(10+T*2,16+L/2
-J,ASC(SEG$(M$,L-J,1)))
190 NEXT J
200 NEXT T
```

Did you ever go through your checkbook 5 times in order to add u your gas bill, then your electric bill, etc.? With this little handy-dandy, you can do it all in one pass.

```
100 CALL CLEAR
110 REM -  ADDER-UPPER by Ji
m Peterson
120 A$="ABCDEFGHIJKLMNOPQRST
UVWXYZ"
130 DIM C$(26),T(26)
140 PRINT "          ADDER-UPP
ER": : :
150 PRINT "WITH THIS PROGRAM
 YOU CAN GO THROUGH YOUR CHE
CKBOOK, OR ANYTHING ELSE, AN
D ADD UP  AMOUNTS IN SEVERA
L CATE-"
160 PRINT "GORIES ALL AT ONE
 TIME.": :
170 PRINT " FIRST, LIST THE
CATEGORIES":"YOU WANT TO ADD
 UP.":" TYPE 'END' WHEN FINI
SHED.": :
180 PRINT " NEXT, ENTER THE
CATEGORY":"CODE AND AMOUNT F
OR EACH":"BILL
```

```
190 PRINT : :"WHEN YOU HAVE
ENTERED ALL":"THE BILLS, TYP
E =": :
200 N=N+1
210 PRINT "CATEGORY #";N
220 INPUT "              ":C$(N
)
230 IF C$(N)="END" THEN 340
240 W$=SEG$(C$(N),1,1)
250 IF POS(A$,W$,1)<>0 THEN
290
260 PRINT :"CODE LETTER ";W$
;" ALREADY  USED - PICK A CO
DE LETTER."
270 INPUT W$
280 GOTO 250
290 X=POS(A$,W$,1)
300 A$=SEG$(A$,1,X-1)&SEG$(A
$,X+1,LEN(A$))
310 X$=X$&W$
320 PRINT :"CODE LETTER FOR
";C$(N);" WILL BE ";W$: :
330 GOTO 200
340 C$(N)=""
350 N=N-1
360 X$=X$&"="
370 IF FLAG=1 THEN 420
380 FLAG=1
390 PRINT : :"READY TO START
 - ": : :
400 PRINT "WHEN FINISHED, TY
PE =": :
410 INPUT "DO YOU WANT TO VE
RIFY EACH  INPUT? ":V$
420 PRINT :"CODE (";X$;")"
430 INPUT Q$
440 IF Q$="=" THEN 600
450 IF POS(X$,Q$,1)<>0 THEN
510
460 PRINT "THAT IS NOT ONE O
F THE CODES": :
470 INPUT "IS IT A NEW CATEG
ORY?(Y/N) ":Q$
480 IF SEG$(Q$,1,1)<>"Y" THE
N 420
490 X$=SEG$(X$,1,LEN(X$)-1)
500 GOTO 200
510 Y=POS(X$,Q$,1)
520 INPUT "AMOUNT ?":A
530 IF SEG$(V$,1,1)="N" THEN
 580
540 PRINT :C$(Y);A: :
550 INPUT "CORRECT? (Y/N)":L
$
560 IF SEG$(L$,1,1)="Y" THEN
 580
570 IF SEG$(L$,1,1)="N" THEN
 420 ELSE 550
580 T(Y)=T(Y)+A
590 GOTO 420
600 FOR J=1 TO N
610 PRINT :C$(J);T(J)
620 TT=TT+T(J)
630 NEXT J
640 PRINT :"GRAND TOTAL OF A
LL IS";TT
650 END
```

And, did you ever wish that you could make numbers smaller, so th you could squeeze more of them on a chart or graph? The problem is that resolution is so poor, at le on my TV screen, but maybe you'll find a use for this.

```
100 REM - NUMBER SCRUNCHER -
 programmed by Jim Peterson
110 CALL SCREEN(5)
120 FOR S=2 TO 14
130 CALL COLOR(S,15,1)
140 NEXT S
150 CALL CLEAR
160 RANDOMIZE
170 DATA 75557,22222,25127,6
1216,55571,74616,74757,71222
,75257,75711
180 FOR J=0 TO 9
190 READ C$
200 CH$(J)="00"&C$
210 NEXT J
220 CH=91
230 INPUT "NUMBER? ":RX
240 N$=STR$(RX)
250 IF LEN(N$)/2=INT(LEN(N$)
/2)THEN 270
260 N$="0"&N$
270 FOR J=1 TO LEN(N$)STEP 2
280 P1=VAL(SEG$(N$,J,1))
290 P2=VAL(SEG$(N$,J+1,1))
300 FOR T=1 TO 7
310 Z$=Z$&SEG$(CH$(P1),T,1)&
SEG$(CH$(P2),T,1)
320 NEXT T
330 CALL CHAR(CH,Z$)
340 Z$=""
350 P$=P$&CHR$(CH)
360 CH=CH+1
370 NEXT J
380 PRINT N$;" ";P$
390 P$=""
400 N$=""
410 GOTO 230
```

Almost OUT OF MEMORY.

Happy hackin'

Jim Peterson

From Sidney News Digest

LOGO EMBLEM

by MIKE SLATTERY

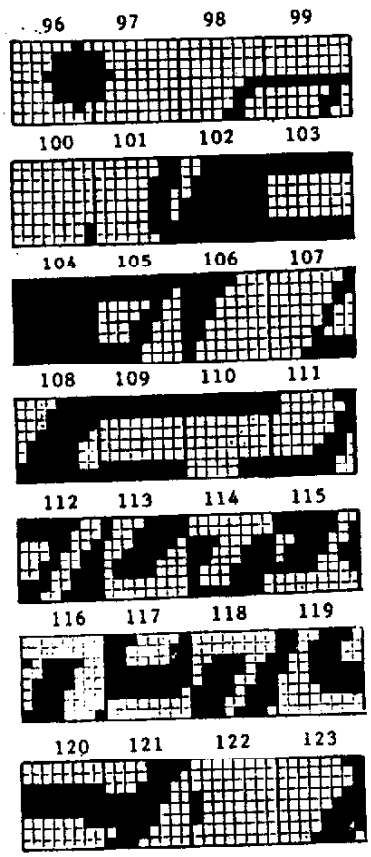
Here is an interesting LOGO design for all those with TI LOGO to enter. Start by defining the tiles shown adjacent using the "MAKECHAR N" option where N in this program is 96 to 143. Each tile is defined separately. When the tile has been defined on the screen, use SHIFT BACK to store the tile in memory.Then call the next tile for defining.

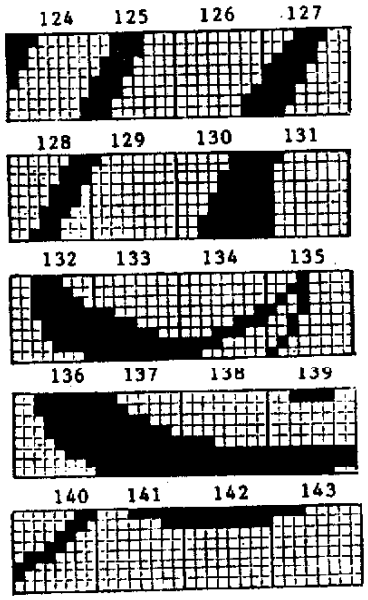
When all tiles have been defined, type in the following program using the procedure option "TO (name)". I have used DESIGN as the procedure name. Type in "TO DESIGN", press ENTER and when the screen shows

```
TO DESIGN
END
```

press ENTER again and type in the program. Be sure to leave spaces between the letter and number groups otherwise you will get an error message.

When the program has been typed in, press SHIFT BACK and then type DESIGN and press ENTER. The design will then be drawn on the screen. Try changing both the foreground and background colors. Can you get the program to change colors while running.






```
TO DESIGN
CS CB 2
PT 96  14 9     PT 97  15 9
PT 98  13 10    PT 99  14 10
PT 100 10 11    PT 101 10 12
PT 102 11 11    PT 103 11 12
PT 104 12 11    PT 105 12 12
PT 106 13 11    PT 107 13 12
PT 108 14 11    PT 109 14 12
PT 110 15 11    PT 111 15 12
PT 112 16 11    PT 113 16 12
PT 114 17 11    PT 115 17 12
PT 116 18 11    PT 117 18 12
PT 118 19 11    PT 119 19 12
PT 120 20 11    PT 121 20 12
PT 122 21 11    PT 123 11 13
PT 124 12 13    PT 125 13 13
PT 126 10 14    PT 127 11 14
PT 128 12 14    PT 129 13 14
PT 130 10 15    PT 131 11 15
PT 132 12 15    PT 133 13 15
PT 134 14 15    PT 135 15 15
PT 136 10 16    PT 137 11 16
PT 138 12 16    PT 139 13 16
PT 140 14 16    PT 141 11 17
PT 142 12 17    PT 143 13 17
END
```

FOR SALE

CONSOLE, PEB (32K, RS-232, DSDD DISK DRIVE, TI DISK CONTROLLER). $400.00

BMC MONITOR $200.00

SIGNALMAN MK IV MODEM $50.00

MUCH SOFTWARE!! XBASIC, MINI-MEMORY, EDITOR/ASSEMBLER, TI-WRITER, MULTIPLAN AND MUCH MORE!!
PRICES ON SOFTWARE NEGOTIABLE.

CONTACT DAVE ELDRIDGE
1-493-6025
(FORT COLLINS)

FOR SALE OR TRADE

HOUSEHOLD MANAGEMENT CARTRIDGE

WANT TO BUY

SPEECH SYNTHESIZER, AND PERSONAL REPORT GENERATOR

CONTACT ED JEFFRESS
344-0079

TI-WRITER

To change RS232.LF to read PIO.LF etc.
submitted by Greg Kimball

INSTRUCTIONS:

1. Forth disk in DSK1.
2. TI-Writer in DSK2.
3. Install Editor/Assembler Cartridge
4. Main Menu: select 2
5. Editor/Assembler Menu: select 3 (LOAD & RUN)
6. DSK1.FORTH in Response
7. FORTH Menu: select Editor
8. Enter: 112 EDIT
9. Top line of the screen displays the RS232.LF
   Type over with PIO.LF or whatever you want,
   and blank out all the extra characters
10. PS. To get the PREVIOUS screen: press FCTN 6
    To get the NEXT screen: press FCTN 4
    To access characters 38-64 (Change Windows)
    press FCTN 5
11. From step 9 press FCTN 9 (For command mode)
12. Enter: 180 DISK_HI !
13. Enter: FLUSH (Writes change to disk)
    (Rem to remove Write Protect)

Rocky Mountain 99'ers

TIC TALK

This publication is printed monthly for the benifit of the membership of the Rocky Mountain 99'ers Computer Club. The Club and the paper are not for the benifit nor backed by any commercial enterprize. Both are non-profit in nature and are for the sole purpose of computer education. Any fees collected are used to defray any cost to maintain the organization. Neither the paper nor the Club have any affiliation with Texas Instruments. Any statements published in this paper are not necessarily the opinion of the membership.

`` OFFICERS and CHAIRMEN ''

PRESIDENT	TED MICHELSEN	986-3513
VICE PRESIDENT	MIKE HOLMES	751-7945
SECRETARY	MARTHA WEEG	320-5589
TREASURER	KEN MONSON	233-1788
EDITOR	DAVID OWEN	458-7315
LIBRARIAN	PETE CROWELL	750-5949
MEMBERSHIP	MARTHA WEEG	320-5589
PROGRAM CHAIRMAN	MIKE HOLMES	751-7945
EDITOR/ASSEMBLER..SIG	MIKE HOLMES	751-7945
TI FORTH.....SIG	PETE CROWELL	750-5949
MULTIPLAN.....SIG	BEN KRAMER	237-1056
THE STAR BOARD....BBS		455-3113

* * ROCKY MOUNTAIN 99ers * *
P.O. Box 12605
Denver, CO 80212

FIRST CLASS

****
* Do you see stars on the label *
* this means your membership is *
* now due. Send in your renew- *
* al today so you don't miss a *
* single issue of TIC-TALK!!! *
****

Dallas TI Home Computer Grp
1221 Mosswood
Irving TX 75061