7/9

BYTE-LINE JULY 1986

NEWSLETTER OF THE DECATUR 99ER USERS' GROUP

INSIDE THIS ISSUE---

PRESIDENT'S NOTES...by Jay Seaberg

Vacation time has arrived again, and many of our members are putting their computing on the back burner. In keeping with the spirit of things, I'll keep this short and sweet. As was mentioned in the last newsletter, it is time to elect officers for the coming year.

Although we were almost trampled by volunteers, the present members of the board prevailed. Our constitution prohibits the President and Vice-President from suceeding themselves. Therefore, the ballot for the election will be as follows:

President	Larry Livergood
Vice-President	Jay Seaberg
Treasurer	Jerry Brunson

Write-in candidates are welcome

Even though summer puts a lot of pressure on our time, let's all make an effort to come to the next meeting. In addition to the election, we need to get together and discuss the direction of the User's Group.

The next meeting will be held on July 10, at 6:30 pm in the Library Board Room. Please try to make it to the meeting. If you can't, then take time to contact one of the officers and make your computing wants and needs known to us.

Even though it is summertime, and livin' may or may not be easy, making use of this computer group is one of the easier things you can do. Simply come to a meeting, or contact one of the group's officers or librarians. You can get software, hardware help, and programming advice from any of these sources.

*** Tom Freeman ***

DOUBLE COLUMN and SIDEWAYS, A REPRISE

Since last summer, when I wrote articles on extended basic programs to print double column text, and to print text sideways (for spreadsheets) using the graphics capabilities of the printer, I have had several requests for a reprint. There were in addition some errors which crept into the printing process, and many people never picked these up in later issues of TOPICS, so I got quite a few "HAALP!" calls. This month's column is a response to these requests - however so as not to completely bore you I have added several enhancements to each program (hence they will be a little longer to type in).

DOUBLE COLUMN

This program makes use of the TI-WRITER formatter to give your text the proper margins and right justify it. The latter is not necessary, but I think it gives the final double column format a neater look. The special formatter commands for underscoring, overstriking and required spacing (&, @, and ^) are preserved, AND ALL SPECIAL CODES YOU SEND TO YOUR PRINTER will also be formatted correctly.

You should begin your text with a line something like this: .LM0;RM56;FI;AD;IN+5;PL300<CR> [Note: <CR> represents the carriage return in TI-WRITER, from now on referred to as TIW, but I can't print the little character for you! The left margin should be 0, but the right margin can be whatever you wish. It will correspond to the column width (minus 1) entered in line 140 of the program. FI(ll) is also necessary, AD(just) is optional, as is IN(dent). The PL (page length) should be greater than the final number of lines in your file, since we do NOT want form feeds to creep in! In determining your right margin remember that TWO columns will be printed side by side, and that you need to compensate for the margins left and right you wish on the final printout, as well as space between.

When you have finished your text, save it using SF, and go to the Formatter. BE SURE that all true "&"s and "@"s have been doubled, as the formatter considers single ones to be formatting commands. "^" needs to be transliterated. Also be careful of an "*" followed by a number (the formatter appears to consider this a command for a mail list). Now for your output file DO NOT use the printer, but instead another disk file, with a different name, e.g. if the input file is DSK1.FILE then the ouput could be DSK1.FILE1. When this process is finished, return to the Editor and load this new file. The first thing you see will be three or more little <LF>'s, and possibly a <PA>. Delete them with FCTN 3. Now scan down the file to the end - you will then see a LOT of <LF>'s depending on how far you were from 300 lines. Delete from here to the end (easiest way is: FCTN 9, D <ENTER>, number of first line to be deleted, comma, E <ENTER>. Note that all the right margins are lined up if you used AD. If you used any printer commands (see end of article) they won't, and you will have to scan the file to see if any wound up at the beginning or end of a line. Once you have finished modifying the file, save it again, using the PF function this time. You may use the same or another name.

Now run the following program in XBasic:

```
100 CALL CLEAR :: DIM A$(200),C(200):: E$=CHR$(27)
:: CR$=CHR$(13):: LF$=CHR$(10):: FF$=CHR$(12):: T$
=CHR$(9):: LT$=CR$&T$ :: PG=1
110 DISPLAY AT(6,1):"DOUBLEPRINT": : :"INPUT FILE?
":"DSK": :"PRINTER NAME?":"PIO" :: ACCEPT AT(10,4)
SIZE(12)BEEP:F$ :: OPEN #1:"DSK"&F$,INPUT :: ACCEP
T AT(13,1)SIZE(-28)BEEP:P$
120 OPEN #2:P$&".CR"
130 DISPLAY AT(1,1)ERASE ALL:"IN THE NEXT 3 INPUTS
,BE SURETHAT TWO TIMES WIDTH LEFT   MARGIN + SPACE
 BETWEEN DOES NOT EXCEED YOUR PRINTER'S   CAPACITY
"

140 DISPLAY AT(7,1):"HOW MANY SPACES LEFT MARGIN?6
": :"HOW MANY BETWEEN COLUMNS?   6": :"WIDTH OF CO
LUMN? 57"
150 ACCEPT AT(8,1)SIZE(-2)BEEP:LEFT :: ACCEPT AT(1
1,1)SIZE(-2)BEEP:BETW :: ACCEPT AT(13,18)SIZE(-2)B
EEP:WIDTH
160 LEFT=LEFT+1 :: RIGHT=LEFT+BETW+WIDTH
170 PRINT #2:CHR$(15);E$;"D";CHR$(LEFT);CHR$(RIGHT
);CHR$(0);!SET CONDENSEDPRINT, TABS
180 DISPLAY AT(15,1):"DO YOU WISH TO RESET LINE
SPACING, COLUMN LENGTH, AND PAGE LENGTH AT EACH PA
GE?   (Y/N) N"
190 ACCEPT AT(18,7)SIZE(-1)VALIDATE("YN")BEEP:AN$
:: IF AN$="Y" THEN CLFLG=1
200 GOSUB 390
210 PRINT #2:E$;"A";CHR$(LS);E$;"C";CHR$(PL)
220 IF EOF(1)THEN CLOSE #1 :: CLOSE #2 :: STOP ELS
E X,Y,X1=0
230 X=X+1 :: LINPUT #1:A$(X):: B=POS(A$(X),LF$,1):
: IF B THEN A$(X)=SEG$(A$(X),1,B-1):: Y=Y+1 :: C(X
)=0 ELSE C(X)=1
240 PRINT X;Y
250 IF X1 THEN 270
260 IF Y=CL THEN X1=X
270 IF Y<2*CL AND EOF(1)=0 THEN 230
280 IF Y<2*CL THEN CLOSE #1 :: GOTO 310
290 GOSUB 350 :: IF CLFLG THEN 300 ELSE 220
300 CALL CLEAR :: PG=PG+1 :: DISPLAY AT(20,11):"PA
GE";PG :: GOTO 200
310 A$(X+1)="" :: EX=0 :: FOR Z=1 TO X :: EX=EX+C(
Z):: IF Z-EX=INT((Y+1)/2)THEN X1=Z :: GOTO 330
320 NEXT Z
330 GOSUB 350
340 CLOSE #2 :: STOP
350 X=0 :: Y=X1
```

```
360 X=X+1 :: PRINT #2:T$;A$(X):: IF C(X)THEN PRINT
 #2:CR$ :: GOTO 360
370 Y=Y+1 :: PRINT #2:T$;A$(Y):: IF C(Y)THEN PRINT
 #2:LT$ :: GOTO 370
380 PRINT #2:LF$ :: IF X<X1 THEN 360 ELSE PRINT #2
:FF$ :: RETURN
390 DISPLAY AT(22,1):"LINES PER COLUMN? 55":"LINE
SPACING 12/72 IN.":"PAGE LENGTH (LINES)? 66"
400 ACCEPT AT(22,19)SIZE(-2)BEEP:CL :: ACCEPT AT(2
3,14)SIZE(-2)BEEP:LS :: ACCEPT AT(24,22)SIZE(-2)BE
EP:PL :: RETURN
```

A quick explanation: Lines 110 to 160 set up the parameters for formatting your printout. You should of course modify the defaults to whatever you most often use, since you then need press only enter at each input. Line 170 then sends the codes to the printer for condensed print and the tabs for each column. Consult your printer manual to confirm that 15, and Escape "D" are the proper codes. The subroutine at line 390 will allow you to input line spacing other than the standard 1/6 inch and set the number of lines per page as well as the printed page length. Lines 180-190 first allow you to signal that you may wish to change these at each page. This is convenient if your text is long, but you wish to put different amounts on each page, or squeeze more in, etc. Line 210 sends the codes for line spacing and page length to your printer (Escape "A" and Escape "C" - check your manual). Line 220 resets parameters for each page, if more is left to do. Now the meat of the program. Line 230 picks up the lines one by one and puts them into the array A$(). An <LF> is checked for (if one is not there the formatter has something special to do, such as underscore or overstrike). Variable Y keeps track of the number of PRINTED lines, X the number of INPUT lines, and the array C() signals which input lines are not to be line fed. Line 240 puts X and Y on the screen, for your interest and can be deleted. X1 will represent the number of input lines to be printed up to the midpoint (i.e. first column) and is determined by line 260. Line 250 skips over this if X1 has already been found. Line 270 returns us for more input, if the end of the file hasn't been reached, and there is still more to do on the page (2*CL lines). If the end of the file has been reached before there were 2*CL lines, then a special routine is needed to caluculate the midpoint (line 310). Otherwise the subroutine at line 350 does ALL the printing. Here X and Y keep track of the left and right columns, and the array C() signals whether the printer should go to the next tab, or return to the same one for an extra line. After all the lines have been printed, there is a form feed and we go back for more (after resetting line spacing and page length if necessary). The last segment in lines 310-340 is for the last, incomplete page and ends in a stop. You could also have it return to line 110.

If your printer doesn't have tab settings, you are almost out of luck, but not quite. It is possible that any special codes you send to your printer won't work - I'm not sure. In any case we take advantage of the computer's tab function. I believe the open statement has to include a VARIABLE 136, or some such. Then make the following substitutions:

```
360 X=X+1 :: PRINT #2:TAB(LEFT);A$(X);CR$:: IF C(X)
THEN 360
370 Y=Y+1 :: PRINT #2:TAB(LEFT+RIGHT);A$(Y);CR$:: IF
C(Y) THEN 370
```

and delete everything in line 170 after CHR$(15).

If you are embedding control codes to your printer in the text, and also wish to right-justify, there is a problem, i.e. the formatter counts them as characters, but the print-head doesn't move, so the right margins won't line up! The following method will compensate: use a <u>single</u> unused character and transliterate it to the sequence of control codes you wish, and <u>add a 32</u> which is a space and will move the print-head the one space that the formatter thought was there. E.g. I used the left brace for underlining .TL 123:32,27,45,1 The 32 should precede the control codes for an opening command and follow them for a closing one. The only problem is when these codes come at the beginning or end of a line, then the spaces aren't buried correctly! The only way to solve this is to scan the formatted file on disk and see whether any do in fact appear at the start or end of a line. In these cases, <u>in fixed mode,</u> delete a space at the beginning and insert it somewhere else, and if the extra space is at the end, insert an extra anywhere on the line (this moves the last character to the end).

<u>SIDEWAYS</u>

This program is deceptively simple. All it requires is a printer capable of dot graphics, and most seem to be these days. The usual code is Escape "K" and is the only one the program can use. The program has been revised to allow for variable width lines, and the data creating program has been extensively revised to allow you to have a CHARA1 file on disk and use that for the character definition. No need to type in a lot of CALL CHAR statements!

The actual sideways printing program merely sets up an array of 60 lines (each block letter is 8 dots wide and 480 dot columns are allowed on a page) and then picks one letter at a time off each one. All it needs is a definition of what the letter looks like on its side, and that is the purpose of the data creating program. It uses the hex codes that TI has already built into the computer for each character and sets up a data statement in a MERGE format. This is then merged, once only, into the final program and that's all there is. I can't just give you the data statements to type in, because they are mostly non printable ASCII codes.

For the data creating program, I originally used the characters built into the console, and suggested typing in a LOT of CALL CHAR statements if you wanted others. The version that follows uses data statements to poke an assembly language program directly into memory. This subprogram looks for a file called CHARA1 on DSK1 and if it is there loads it into the pattern descriptor table in VDP ram (it also overwrites the color table and you will see crazy things on the screen until the program is finished!) The AL program requires a DSRLNK for disk access. In XBasic this would have required a LOT of extra typing since the routine doesn't exist in the module. Editor/Assembler does have it. Hence the following program is in Basic and MUST be run with the E/A module in.

```
 100 DATA 5,0,2,250,0,0,8,0,0,11,68,83,75,49,46,67,
72,65
 110 DATA 82,65,49,0,2,0,16,0,2,1,39,16
 120 DATA 2,2,0,21,4,32,33,16,2,0,16,9,200,0,131,86
,4,32
 130 DATA 33,32,0,8,4,224,131,124,4,91,255
 140 DATA 65,32,32,32,32,32,39,38,255
 150 CALL INIT
 160 X=10000
 170 READ A
 180 IF A=255 THEN 220
 190 CALL LOAD(X,A)
 200 X=X+1
 210 GOTO 170
 220 IF X>10100 THEN 250
 230 X=16176
 240 GOTO 170
 250 CALL LOAD(8234,63,48)
 260 CALL LINK("A")
 270 OPEN #1:"DSK1.DATAMERGE",VARIABLE 163
 280 FOR X=1 TO 19
 290 PRINT #1:CHR$(0);CHR$(X*5);CHR$(147);
 300 FOR Y=1 TO 5
 310 CALL CHARPAT(X*5+Y+26,C$)
 320 GOSUB 470
 330 D$=""
 340 FOR Z=8 TO 1 STEP -1
 350 D$=D$&CHR$(D(Z))
 360 NEXT Z
 370 PRINT #1:CHR$(199);CHR$(8);D$;
 380 IF Y=5 THEN 390 ELSE 410
 390 PRINT #1:CHR$(0)
 400 GOTO 420
 410 PRINT #1:CHR$(179);
 420 NEXT Y
 430 NEXT X
 440 PRINT #1:CHR$(255)&CHR$(255)
 450 CLOSE #1
 460 STOP
 470 FOR Z=1 TO 8
 480 E1$=SEG$(C$,2*Z-1,1)
 490 E2$=SEG$(C$,2*Z,1)
 500 F1=ASC(E1$)-48+7*(ASC(E1$)>60)
 510 F2=ASC(E2$)-48+7*(ASC(E2$)>60)
 520 D(Z)=F1*16+F2
 530 NEXT Z
 540 RETURN
```

You now have a DISPLAY, VARIABLE 163 file on the disk in DSK1 named DATAMERGE. Now type in the following program which will become the main program. Note that in line 150 you can set the maximum width of a line if it is less than 80 characters, and that line 160 resets the length of page to one more line than this, so that contiguous spreadsheets can be printed together. Check your printer manual for lines 130 and 160. When you have finished typing the program as printed in this article, then type in MERGE DSK1.DATAMERGE. This will put in 19 lines from 5 to 95. <u>Do not list</u> the program in this form to a printer, because lines 5-95 can't be handled by a printer.

SIDEWAYS is now ready to use. The only limitation is that the files on disk to be used must contain <u>only</u> ASCII characters 32-127, and must have been saved using the PF function of the TIW, not SF.

```
 100 CALL CLEAR :: PRINT "SIDEWAYS PRINT ":" by Tom
 Freeman": :
 110 DIM D$(126),A$(60):: FOR X=32 TO 126 :: READ D
 $(X):: NEXT X
 120 P$="PIO.CR"
 130 ESC$=CHR$(27):: OPEN #2:P$ :: PRINT #2:ESC$&"A
 "&CHR$(7)!RESET LINE FEED TO 7 DOTS(7/72 IN)
 140 FLAG=0 :: PRINT "TEXT FILE:":" DSK";:: INPUT "
 ":F$ :: ON ERROR 140 :: OPEN #1:"DSK"&F$,INPUT ::
 ON ERROR STOP
 150 INPUT "MAX LINE WIDTH <81 ":Z :: IF Z>80 THEN
 150
 160 PRINT #2:ESC$;"C";CHR$(Z+1)!RESET FORM FEED TO
  Z+1 LINES
 170 FOR X=1 TO 60 :: LINPUT #1:A$(X):: PRINT A$(X)
 :: IF EOF(1)THEN 190
 180 NEXT X :: GOTO 210
 190 FLAG=1 :: CLOSE #1 :: IF X=61 THEN 210
 200 FOR X=X+1 TO 60 :: A$(X)=RPT$(" ",Z):: NEXT X
 210 FOR X=1 TO 60 :: A$(X)=A$(X)&RPT$(" ",Z-LEN(A$
 (X))):: NEXT X :: FOR X=1 TO Z :: PRINT #2:ESC$&"K
 "&CHR$(224)&CHR$(1):: PRINT "PRINTER LINE";X
 220 FOR Y=60 TO 1 STEP -1 :: B$=SEG$(A$(Y),X,1)
 230 PRINT #2:D$(ASC(B$));
 240 NEXT Y :: PRINT #2:CHR$(13)&CHR$(10):: NEXT X
 :: PRINT #2:CHR$(12):: IF FLAG=0 THEN 170
 250 INPUT "DO ANOTHER?(Y/N)":AN$ :: IF AN$="Y" THE
 N 140
```

<u>VARIABLE COLUMN PROGRAM LISTER</u>

This program will take any basic or XBasic program which has been listed to disk (e.g. LIST"DSK1.PROGRAM") and reformat it to ANY column width you want. The meat of the program is in lines 180-280 plus subroutine 320. The rest is bells and whistles, enabling you to set a left margin, double space between PROGRAM lines (not printer lines), and set up any printer codes you want. Defaults are retained if you go back for another listing. In case you have reset the printer control codes, a RESET is printed via Escape "@" in line 160. If your printer does not use this method, you'll have to do it some other way.

Because the program checks for a <u>number</u> at the 81st and 161st characters in order to see if the next record in the file is actually a new line number, a possible error arises if a number happens to be there which is both an allowable line number (1-32767) and higher than the previous line number. Then the program assumes it is a new line and splits it off. Most of these errors can be avoided if your program to be listed has been REsequenced with a regular interval, and you indicate such at the prompt.

Now a brief explanation of the program. Line 180 picks up a record which may be all or part of a program line, and sets a flag indicating something has been started. Line 190 is the key to the program. The VAL function in XBasic will give an actual number if the characters up to the first space in the string are numbers only. This is what we want to see at the beginning of a program line, i.e. a line number. The ON ERROR 420 statement keeps the program from crashing if this is not a valid number. Instead it makes the program go on and will treat the string as the continuation of a line. Line 200 says that if the length of the string is less than 80 than surely the end of a (listed) program line has been reached, and the subroutine at 320 goes to print it, splitting it if necesary and putting in the margins selected if any. If the length WAS exactly 80 then we go for another record and again check for a valid number at the beginning, and if there is one whether it could be a line number (using either the allowable range or the increment). If the number doesn't fit into one of these categories the string is joined to the previous one and we go on.

Note the use of the following subroutines. 380-410 takes a string of ASCII numbers separated by spaces and concatenates them into a single string suitable for sending to the printer. 430-470 is a simulated cursor which allows the use of the enter key but does not require it, allowing the simple press of the Y or N key as well.

That's all there is to it. The program seems to process the records as fast as my printer can handle them. Note that if you use a disk file as your output device, as I did to produce the program listings in these articles, then the file as listed on the disk will be DISPLAY, VARIABLE X where X is the sum of your listing width and the margin. You cannot load this into TI-WRITER directly. You must use a sector editor such as Advanced Diagnostics to change byte 17 (hex >11) of the file descriptor record (catalog sector) from whatever is there to >50. This also means that you can't use widths that exceed 80 columns directly in TIW.

```
100 O$="PIO" :: W=136 :: P$="15" :: INC=10
110 DISPLAY AT(2,4)ERASE ALL:"VARIABLE COLUMN LIST
ER": :"    by Tom Freeman": :"INPUT LISTFILE?":"D
SK";F$: :"OUTPUT DEVICE?":O$: :"MAXIMUM LINE WIDTH
?";W: :"LEFT MARGIN?";M
120 ACCEPT AT(7,4)SIZE(-28)BEEP:F$ :: ACCEPT AT(10
,1)SIZE(-28)BEEP:O$ :: ACCEPT AT(12,21)SIZE(-3)BEE
P:W :: ACCEPT AT(14,14)SIZE(-3)BEEP:M
130 DISPLAY AT(16,1):"PRNTER CTRLS?(SEP.BY SPACES)
";P$ :: DISPLAY AT(20,1):"LINE NUMBER INCREMENT(O
IF  VARIABLE)";INC: : :"DOUBLE SP?  (Y/N) Y"
140 ACCEPT AT(17,1)SIZE(-28)VALIDATE(DIGIT," ")BEE
P:P$ :: ACCEPT AT(21,11)SIZE(-3)BEEP:INC :: CT$=""
 :: PR$=P$ :: GOSUB 430 :: IF K=89 THEN SP=1 ELSE
SP=0
150 DISPLAY AT(24,1)BEEP:"ALL CORRECT(Y/N)? Y" ::
GOSUB 430 :: IF K=78 THEN 120
160 GOSUB 380 :: OPEN #1:"DSK"&F$ :: OPEN #2:O$,VA
RIABLE W+M :: M$=RPT$(" ",M):: STOPFLAG=0 :: PRINT
 #2:CHR$(27);"@";CT$;
170 GOSUB 290 :: IF STOPFLAG=1 THEN 370
180 ON ERROR STOP :: LINPUT #1:A$ :: FLAG=1
190 ON ERROR 420 :: L1=VAL(SEG$(A$,1,POS(A$," ",1)
-1))
200 IF LEN(A$)=80 THEN 220
210 GOSUB 320 :: FLAG=0 :: GOTO 170
220 GOSUB 290 :: IF STOPFLAG=1 THEN 370
230 LINPUT #1:B$ :: A=POS(B$," ",1):: IF A=0 THEN
280
240 ON ERROR 280 :: L2=VAL(SEG$(B$,1,A-1)):: IF IN
C THEN 260
250 IF L2>32767 OR L2<=L1 THEN 280 ELSE 270
260 IF L2<>L1+INC THEN 280 ELSE 270
270 GOSUB 320 :: A$=B$ :: GOTO 190
280 A$=A$&B$ :: IF LEN(B$)<80 THEN 210 ELSE 220
290 IF EOF(1)THEN CLOSE #1 ELSE RETURN
300 IF FLAG THEN GOSUB 320
310 CLOSE #2 :: STOPFLAG=1 :: RETURN
320 IF M=0 THEN PRINT #2:A$ :: GOTO 350
330 L=LEN(A$):: IF L THEN R$=SEG$(A$,1,W)ELSE 350
340 PRINT #2:M$&R$ :: A$=SEG$(A$,W+1,255):: GOTO 3
30
350 IF SP=1 THEN PRINT #2
360 RETURN
370 DISPLAY AT(24,1)BEEP:"DO ANOTHER (Y/N)? Y" ::
GOSUB 430 :: IF K=89 THEN 110 ELSE STOP
380 A=POS(PR$," ",1):: B=LEN(PR$):: IF B=0 THEN RE
TURN
390 IF ASC(PR$)=32 THEN PR$=SEG$(PR$,2,28):: GOTO
380
400 IF A=0 THEN CT$=CT$&CHR$(VAL(PR$)):: RETURN
410 CT$=CT$&CHR$(VAL(SEG$(PR$,1,A))):: PR$=SEG$(PR
$,A+1,28):: GOTO 380
420 RETURN 200
430 CALL KEY(0,K,S):: IF S=-1 THEN 430 ELSE IF K=1
3 OR K=78 OR K=89 THEN 460
440 IF Y=89 THEN Y=30 ELSE Y=89
450 CALL HCHAR(24,21,Y):: GOTO 430
460 IF K=13 THEN K=89
470 CALL HCHAR(24,21,K):: RETURN
```

QUAD COLUMN

by Tom Freeman

This is yet another version of Double Column! It was written by popular demand ("if you can do two, why not four!") but wa_ c quite ready when last month's news letter was printed. What follows is Double Column exactly, with all changes or additions underlined. This is to make it easier if you have the previous version already typed in. Resequence it for neatness when you have it fully debugged. The instructions remain the same, so refer please to TopIcs May, 1986.

```
100 OPTION BASE 1 :: CALL CLEAR :: DIM A$(300),C(300),COL(4):: E$=CHR$(27)::
 CR$=CHR$(13):: LF$=CHR$(10):: FF$=CHR$(12):: T$=CHR$(9):: LT$=CR$&T$ :: LT1
$=LT$&T$ :: LT2$=LT1$&T$ :: PG=1
110 DISPLAY AT(6,1):"DOUBLEPRINT": : :"INPUT FILE?":"DSK": :"PRINTER NAME?":
"PIO" :: ACCEPT AT(10,4)SIZE(12)BEEP:F$ :: OPEN #1:"DSK"&F$,INPUT :: ACCEPT
AT(13,1)SIZE(-28)BEEP:P$
120 OPEN #2:P$&".CR"
130 DISPLAY AT(1,1)ERASE ALL:"IN THE NEXT 3 INPUTS,BE SURETHAT TWO TIMES WID
TH LEFT   MARGIN + SPACE BETWEEN DOES NOT EXCEED YOUR PRINTER'S  CAPACITY"
140 DISPLAY AT(7,1):"HOW MANY SPACES LEFT MARGIN?6": :"HOW MANY BETWEEN COLU
MNS?   6": :"WIDTH OF COLUMN? 57": :"NUMBER OF COLUMNS? 2"
150 ACCEPT AT(8,1)SIZE(-2)BEEP:LEFT :: ACCEPT AT(11,1)SIZE(-2)BEEP:BETW :: A
CCEPT AT(13,18)SIZE(-2)BEEP:WIDTH :: ACCEPT AT(15,20)SIZE(-1)VALIDATE("234")
BEEP:COLUMN
160 LEFT=LEFT+1 :: DIFF=BETW+WIDTH :: RIGHT=LEFT+DIFF :: FOR X=2 TO COLUMN :
: COL(X)=1 :: NEXT X
170 PRINT #2:CHR$(15);E$;"D";CHR$(LEFT);CHR$(RIGHT);CHR$(-(COL(3)=1)*(RIGHT+
DIFF));CHR$(-(COL(4)=1)*(RIGHT+2*DIFF));CHR$(0);!SET COND,TABS
180 DISPLAY AT(17,1):"DO YOU WISH TO RESET LINE   SPACING, COLUMN LENGTH, AN
D PAGE LENGTH AT EACH PAGE?   (Y/N) N"
190 ACCEPT AT(20,7)SIZE(-1)VALIDATE("YN")BEEP:AN$ :: IF AN$="Y" THEN CLFLG=1
200 GOSUB 390
210 PRINT #2:E$;"A";CHR$(LS);E$;"C";CHR$(PL)
220 IF EOF(1)THEN CLOSE #1 :: CLOSE #2 :: STOP ELSE X,Y,X1,X2,X3=0
230 X=X+1 :: LINPUT #1:A$(X):: B=POS(A$(X),LF$,1):: IF B THEN A$(X)=SEG$(,  (
X),1,B-1):: Y=Y+1 :: C(X)=0 ELSE C(X)=1
240 PRINT X;Y
250 IF X1 THEN 261
260 IF Y=CL THEN X1=X :: GOTO 270
261 IF X2 AND COL(3)THEN 265
262 IF Y=2*CL THEN X2=X :: GOTO 270
265 IF X3 AND COL(4)THEN 270
266 IF Y=3*CL THEN X3=X
270 IF Y<COLUMN*CL AND EOF(1)=0 THEN 230
280 IF Y<COLUMN*CL THEN CLOSE #1 :: GOTO 310
290 GOSUB 350 :: IF CLFLG THEN 300 ELSE 220
300 CALL CLEAR :: PG=PG+1 :: DISPLAY AT(20,11):"PAGE";PG :: GOTO 200
310 A$(X+1),A$(X+2),A$(X+3)="" :: EX=0 :: FOR Z=1 TO X :: EX=EX+C(Z):: IF Z-
EX=INT((Y+COLUMN-1)/COLUMN)THEN X1=Z :: IF COL(3)=0 THEN 330
311 IF Z-EX=2*INT((Y+COLUMN-1)/COLUMN)THEN X2=Z :: IF COL(4)=0 THEN 330
312 IF Z-EX=3*INT((Y+COLUMN-1)/COLUMN)THEN X3=Z :: GOTO 330
320 NEXT Z
330 GOSUB 350
340 CLOSE #2 :: STOP
350 X=0 :: Y=X1 :: Y1=X2 :: Y2=X3
360 X=X+1 :: PRINT #2:T$;A$(X);:: IF C(X)THEN PRINT #2:CR$;:: GOTO 360
370 Y=Y+1 :: PRINT #2:T$;A$(Y);:: IF C(Y)THEN PRINT #2:LT$;:: GOTO 370
375 IF COL(3)=0 THEN 380 ELSE Y1=Y1+1 :: PRINT #2:T$;A$(Y1);:: IF C(Y1)THEN
PRINT #2:LT1$;:: GOTO 375
376 IF COL(4)=0 THEN 380 ELSE Y2=Y2+1 :: PRINT #2:T$;A$(Y2);:: IF C(Y2)THEN
PRINT #2:LT2$;:: GOTO 376
380 PRINT #2:LF$ :: IF X<X1 THEN 360 ELSE PRINT #2:FF$ :: RETURN
390 DISPLAY AT(22,1):"LINES PER COLUMN? 55":"LINE SPACING 12/72 IN.":"PAG  L
ENGTH (LINES)? 66"
400 ACCEPT AT(22,19)SIZE(-2)BEEP:CL :: ACCEPT AT(23,14)SIZE(-2)BEEP:LS :: AC
CEPT AT(24,22)SIZE(-2)BEEP:PL :: RETURN
```

PROGRAMMERS PAGE SCREEN TITLES

The following three programs are examples of how to show titles or text on the screen. They can be used for program headings, displaying instructions or if you are into VCR's they would allow you to produce some very professional looking results.

Each of the three examples are without color or animation, I will leave it to you to add these features. I have gone over each of the programs and they are copies of working programs. If you type them in and have bugs that you cannot exorcise let one of the officers know, perhaps we can have a debug section some Sunday afternoon. Let me know before a meeting and I will try to help you find it. (Please bring a disk copy of the program, or a cassette copy and your cassette and cables so that we can look at your work, I know the programs as listed work. Also bring a copy of the program.)

The first program SPIRAL is fairly easy to uderstand. The TELETYPE is next in order of difficulty and CENTER BURST is the most difficult. Teletype and Center Burst are contained in one program but should be easy to seperate for actual use.

An example of a graphics MARQUEE can be found on page III-17 of the TI User's Reference guide. This is a very handy book for those of you who do not beleive in cracking books.

SPIRAL

From the MUSIC CITY 99ers Users Group newsletter UPTIME come the following:

Here is a short listing that can be saved in MERGE format for use in your title screens. When run, it gives the appearance od a theatre marquee.

In the listing, R1 is the left side of the box, R2 is the right side, C1 is the top, and C2 is the bottom. The "Z" FOR-NEXT loop is the number of orbits the spiral makes. By changing the values of R1, R2, C1, C2, and the "Z" loop, you can place the spiral anywhere on the screen, make it any size, and cause it to orbit as many times as you like.
(Ed. note: be careful when you type in lines 170 and 180. line number 180 the first time is part of line 170, the second time it is line 180.

```
100 ! ********SPIRAL*******
      * TI Extended Basic *
      *   T. E. Cushing   *
      *   Nashville, TN   *
      ********7/85*********
110 CALL CLEAR :: CALL SCREE
N(16):: CH=42
120 R1,C1=1 :: R2=24 :: C2=3
2 :: FOR Z=1 TO 12
130 FOR A=C1 TO C2 :: CALL H
CHAR(R1,A,CH):: NEXT A :: R1
=R1+1 :: FOR A=R1 TO R2 :: C
ALL VCHAR(A,C2,CH):: NEXT A
:: C2=C2-1
140 CALL KEY(0,K,S):: IF S<>
0 THEN 200
150 FOR A=C2 TO C1 STEP -1 :
: CALL HCHAR(R2,A,CH):: NEXT
 A :: R2=R2-1 :: FOR A=R2 TO
 R1 STEP -1 :: CALL VCHAR(A,
C1,CH):: NEXT A :: C1=C1+1
160 CALL KEY(0,K,S):: IF S<>
0 THEN 200
170 NEXT Z :: IF CH=42 THEN
180 ELSE IF CH=32 THEN 190
180 CH=32 :: GOTO 120
190 CH=42 :: GOTO 120
200 CALL CLEAR :: END
```

To see what it looks like on a title scren, insert the following line:

```
>112 DISPLAY AT(11,12):"THEAT
RE":TAB(12);"MARQUEE" :: DIS
PLAY AT(23,3):"HOLD ANY KEY
TO CONTINUE"
```

Now change line 120 to read:

```
>120 R1=8 :: R2=15 :: C1=11 :
: C2=23 :: FOR Z=1 TO 2
```

The two CALL Key commands at lines 140 and 160 are inserted so that you can get out of the loop and on with the program.

The following article is reprinted from the COMPUTER BRIDGE the newsletter of the ST. LOUIS 99'ERS USERS GROUP.

FANCY TEXT ON THE SCREEN
by Roy T. Tamashiro

Interesting screen presentations can catch the attention of computer users and bystanders alike. The two routines below allow you to display text on the screen in a "fancy", attention getting way. These routines are useful in creating title screens for your programs.

The first routine, CENTER BURST, makes each line of text emerge from the center of the screen (line), as though the letters were emerging from a geyser or a volcano. When the screen is complete, the letters "fall back" into the center of the row. The other routine, TELETYPE, displays text character by character like a teletype machine. The text is centered on each line, and lines are printed both left to right and right to left. The text is wiped from the screen using the same teletype action. Both routines include sound effects for dramatic effect.

PROGRAMMER'S PAGE - SCREEN TITLES CON

To use the routines, type the CENTER BURST and TELETYPE routinesand save them on cassette or disk. They may be typed and saved seperately, or they may be combined into one continous listing. When you want to include either (or both) routine(s) in your program, load the routine(s) into the computer's memory, and then add DATA statements to specify what you want displayed on the screen. The format of the DATA statements is as follows:

(Line number) DATA(Screen Row Number(1-24),(Text)

For Example:

```
250 DATA 1,"My Title Screen"
```

In this example, the message "My Title Screen" will appear on row 1.

Note the following precautions:

(1) The Row Number must be between 1 and 24.

(2) The text may not be longer than 8 characters per row.

(3) Be sure no other program lines follow the CENTER BURST and TELETYPE routines.

After the last line to be displayed on the screen add a final DATA statement with a number larger than 24 to indicate that no further lines are to be displayed on that screen.

For Example:

```
750 DATA 99
```

Use CALL BURST or CALL TELETYPE following your last DATA statement to invoke the proper routine. The FANCY TEXT DEMO listed below illustrates how this routine is carried out.

FANCY TEXT

```
100 !********************
110 !* FANCY TEXT DEMO *
120 !********************
130 !Author: ROY TAMASHIRO
140 !Language: X-BASIC
150 !September 1985
160 DATA 1,"CENTER-BURST TEX
T"
170 DATA 2,"----------------
-"
180 DATA 3,"BY ROY TAMASHIRO
 "
190 DATA 7,"In this routine,
 lines are"
200 DATA 8,"written from the
 center of"
210 DATA 9,"the screen outwa
rd."
220 DATA 22,"Then the lines
are erased"
230 DATA 23,"back into the c
enter."
240 DATA 99
250 CALL BURST
260 DATA 1,"THE TELETYPE MAC
HINE"
270 DATA 2,"----------------
----"
280 DATA 3,"By Roy Tamashiro
"
290 DATA 5,"This routine dis
plays text"
300 DATA 8,"like a Teletype
machine."
310 DATA 9,"Any line 28-char
acters or"
320 DATA 10,"less is centere
d on the"
330 DATA 11,"screen on the r
ow you"
340 DATA 12,"designate."
350 DATA 21,"Then the lines
are erased."
360 DATA 99
370 CALL TELETYPE
380 END
```

```
31000 !******************
31010 !* CENTER BURST *
31020 !******************
31030 !AUTHORL ROY TAMASHIRO
31040 SUB BURST
31050 DIM R$(24)
31060 FOR I=1 TO 24 :: R$(I)
="" :: NEXT I :: CALL CLEAR
31070 READ ROW :: IF ROW<25
THEN READ R$(ROW):: GOTO 310
70
31080 FOR R=1 TO 24 :: IF R$
(R)="" THEN 31160 ELSE W$=R$
(R)
31090 LLF=LEN(W$)/2 :: LRT=L
EN(W$)-LLF
31100 LEFT$=SEG$(W$,1,LLF)::
 RIGHT$=SEG$(W$,LLF+1,LEN(W$
))
31110 CALL HCHAR(R,1,32,32)
31120 FOR I=0 TO LRT
31130 IF LRT-I/1 THEN DISPLA
Y AT(R,15):SEG$(RIGHT$,LRT-I
,I+1)
31140 IF LLF-I THEN DISPLAY
AT(R,14-I)SIZE(I+1):SEG$(LEF
T$,1,I+1)
31150 CALL SOUND(-50,990,1):
: NEXT I
31160 NEXT R
31170 FOR R=1 TO 24 :: IF R$
(R)="" THEN 31300
31180 RIGHT$="" :: LEFT$=""
31190 FOR I=3 TO 16
31200 CALL GCHAR(R,I+14,B)::
 CALL GCHAR(R,I,A)
31210 LEFT$=LEFT$&CHR$(A)::
RIGHT$=RIGHT$&CHR$(B)
31220 NEXT I
31230 FOR I=1 TO LEN(LEFT$)
31240 CALL SOUND(-50,-2,1)
31250 DISPLAY AT(R,15)SIZE(L
EN(RIGHT$)+1):RIGHT$:""
31260 DISPLAY AT(R,I)SIZE(LE
N(LEFT$)+1):LEFT$:""
31270 LEFT$=SEG$(LEFT$,1,LEN
(LEFT$)-1)
31280 RIGHT$=SEG$(RIGHT$,2,L
EN(RIGHT$)-1)
31290 NEXT I :: CALL HCHAR(R
,17,32)
31300 NEXT R
31310 SUBEND
```

```
32000 !*************
32010 !* TELETYPE *
32020 !*************
32030 !AUTHOR:Roy Tamashiro
32040 SUB TELETYPE
32050 DIM W$(24):: CALL CLEA
R
32060 FOR I=1 TO 24 :: W$(I)
="" :: NEXT I
32070 READ ROW :: IF ROW<25
THEN READ W$(ROW):: GOTO 320
70
32080 FOR R=0 TO 23 STEP
32090 IF W$(R)="" THEN 32160
32100 START=INT(17-LEN(W$(R)
)/2)
32110 FOR C=3 TO 30
32120 CALL HCHAR(R,C,30):: C
ALL SOUND(-10,990,1)
32130 IF C>=START AND LEN(W$
(R))>C-START THEN DISPLAY AT
(R,C-2)SIZE(1):SEG$(W$(R),1+
C-START,1):: GOTO 32150
32140 CALL HCHAR(R,C,32)
32150 NEXT C
32160 IF W$(R+1)="" THEN 322
30
32170 START=INT(16+LEN(W$(R+
1))/2):: L1=LEN(W$(R+1))
32180 FOR C=30 TO 3 STEP -1
32190 CALL HCHAR(R+1,C,30)::
 CALL SOUND(-10,990,1)
32200 IF C<=START AND L1>=(S
TART-C+1)THEN DISPLAY AT(R+1
,C-2)SIZE(1):SEG$(W$(R+1),L1
-(START-C),1):: GOTO 32220
32210 CALL HCHAR(R+1,C,32)
32220 NEXT C
32230 NEXT R
32240 FOR R=1 TO 24 STEP 2 :
: IF W$(R)="" THEN 32260
32250 FOR C=3 TO 30 :: CALL
HCHAR(R,C,30):: CALL SOUND(-
50,-2,1):: CALL HCHAR(R,C,32
):: NEXT C :: W$(R)=""
32260 IF W$(R+1)="" THEN 3
90
32270 FOR C=30 TO 3 STEP -1
:: CALL HCHAR(R+1,C,30):: CA
LL SOUND(-50,-2,1):: CALL HC
HAR(R+1,C,32):: NEXT C
32280 W$(R+1)=""
32290 NEXT R
32300 SUBEND
```

SPRITE PROGRAMS

MERGE THAT PROGRAM!

Without reproducing the entire article I would like to cover an article and discovery by Jack and BJ Mathis of the Southwest Ninety-Niners, of Tucson, Az.

While working on a program they started to get error messages where there were none only a short time before. The tried a backup Extended Basic Cartridge, even a backup 99/4A system, nothing seemed to work.

"I vaguely remembered something about the way the CPU stacks the programs in, by puting the last line number entered on top of the stack. The MERGE command reshuffles the program back into proper order. So, I asved the program in MERGE format, typed NEW, and MERGEd the program back in. Then I RESequenced again. No more error codes. It also shortened the program file (less linkage?)."

I have tried this on a number of programs that I have written or worked on and it does seem to help. If the program lines are in order then the computer does not have to wait while it's finding the next line number, as is the case when lines have been added out of sequence. If you try this technique please report what your dings are.

From J. Larry Schott of the West Jax 99ers News, Orange Park, Fl. comes this very interesting demo of a sprite circle.

CALL LOCATE CIRCLE

```
>100 1985 J. LARRY SCHOTT
>200 ! CALL LOCATE CIRCLE
>210 CALL CLEAR :: CALL COLOR
 (8,5,5):: FOR ROW=7 TO 19 ::
 CALL HCHAR(ROW,10,95,13):: N
 EXT ROW :: RADIUS=50
>220 FOR SP=1 TO 25 :: CALL S
 PRITE(*SP,42,2,256,1):: NEXT
 SP
>230 FOR A=0 TO 100 STEP 4 ::
  S=A/100*(2*PI):: X=INT(SIN(
 S)*RADIUS):: Y=INT(COS(S)*RA
 DIUS)
>240 CALL LOCATE(*A/4+1,X+97,
 Y+121):: NEXT A
>250 CALL DELSPRITE(ALL):: RA
 DIUS=INT(RND*86+10):: GOTO 2
 20
```

Explanation:

210 Initializes by drawing a square on the screen as a reference. Also sets the first pass radius.
2 Creates 25 sprites and places them off screen bottom for now.
230 Loop figures 25 points on circle.
240 Places sprites on points of circle, using CALL LOCATE.
250 Erases circle, gets new radius size for variety, then loops back.

After you watch this for awhile, try this: delete the FOR/NEXT loop in 220 and change *SP to *1 there. Change *A/4+1 to *1 in line 240. When you run it like this, you have just one sprite flying around.

ARROW

This is another demonstration of the almost unbeleivable power and capabil-ities of 99/4A sprites in Extended Basic

```
1 !!!!!!!!!!!!!!!!
2 ! BY DANNY COX !
3 !!!!!!!!!!!!!!!!
4 !
5 CALL MAGNIFY(4):: CALL CLE
AR :: CALL SCREEN(2)
6 CALL CHAR(96,"FFFFFFFFFFFF
FFFFFF7F3F1F0F07030180C0E0F0
F8FCFEFFFFFFFFFFFFFFFFFFFF")
7 FOR X=7 TO 4 STEP -1
8 R=190 :: C=250
9 FOR I=1 TO 25 :: CALL SPRI
TE(#I,96,RND*13+3,R,C):: R=R
-X :: C=C-7 :: NEXT I
10 R=190 :: C=250
11 FOR I=25 TO 1 STEP -1 ::
CALL SPRITE(#I,96,RND*13+3,R
,C):: R=R-X :: C=C-7 :: NEXT
 I
12 NEXT X
13 FOR X=5 TO 7
14 R=190 :: C=250
15 FOR I=1 TO 25 :: CALL SPR
ITE(#I,96,RND*13+3,R,C):: R=
R-X :: C=C-7 :: NEXT I
16 R=190 :: C=250
17 FOR I=25 TO 1 STEP -1 ::
CALL SPRITE(#I,96,RND*13+3,R
,C):: R=R-X :: C=C-7 :: NEXT
 I
18 NEXT X :: GOTO 7
```

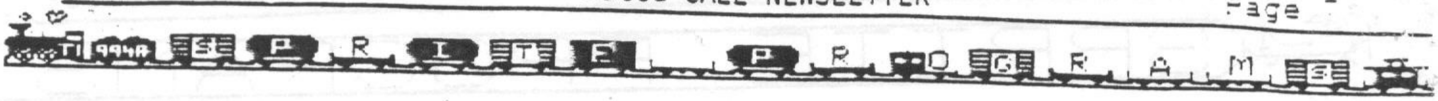
BACKGROUND NIBBLING
from the Corpus Christi 99'ers

The following program is an excersize in doing things backward. I have seen too many rockets blasting off thru the top of my television. This one gives the illusion of movement by nibbling away the background

Notice the use of the RPT$(*) statement, it's the only thing I could think of using it for!

You can write your own games around it. Run it then press any key to start the rise/fall effect.

Mac



```
100 CALL CLEAR :: CALL SCREE
N(2):: FOR I=1 TO 24 :: CALL
 HCHAR(I,1,130,32):: NEXT I
:: CALL COLOR(13,13,13)
110 CALL CHAR(112,"010101030
3070707070F1F3F7FE7C38300000
08080C0C0C0C0E0F0F8FCCE8682"
)
120 CALL CHAR(116,"0103070F1
B13113F2B2B3B2D2D0705010080C
0E0E070B0B05070B0F0D0404000"
)
130 CALL MAGNIFY(3):: CALL S
PRITE(*1,112,15,100,120)
140 CALL KEY(0,K,S):: IF S<>
1 THEN 140 ELSE CALL SPRITE(
*2,116,9,117,120)
150 CALL COLOR(9,2,13):: CAL
L CHAR(96,"",97,"FFFFFFFFFFF
FFFFF")
160 FOR I=1 TO 24 :: CALL CH
AR(96,A$):: CALL HCHAR(I,1,9
6,32):: FOR J=1 TO 8 :: CALL
 CHAR(96,RPT$("FF",J)):: NEX
T J :: CALL HCHAR(I,1,97,32)
:: NEXT I
170 GOTO 170
```

WEEKEND QUICKIE
by Ed Lee

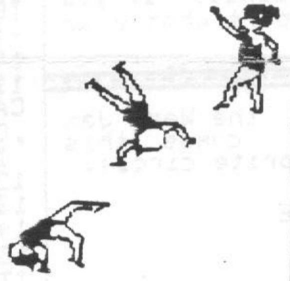
This little quickie comes from The CompuServe TI-Forum. Ron Albright a sysop on the TI-Forum and author of the **ORPHAN CHRONICLES** put it up on the BBS for all to have.

Ron's description is "it emulates the microscopic appearance of some notorious spirochetes".

```
100 !***********
110 !*ORGANISMS*
120 !***********
130 !By: Ed Lee
140 !1985
150 !
160 X=11 :: CALL SCREEN(2)::
 DISPLAY ERASE ALL :: CALL C
HAR(33,"1"):: FOR I=1 TO 28
:: CALL SPRITE(*I,33,X,96,12
8):: NEXT I
170 FOR I=4 TO 28 STEP 4 ::
X=(RND*5+1)*SGN(RND-.5):: Y=
(RND*5+1)*SGN(RND-.5):: FOR
J=I-3 TO I :: CALL MOTION(*J
,X,Y):: NEXT J :: NEXT I
180 GOTO 170
```

SPRITE DEMO

The following program is an example of sprite animation using data statements and a loop to control the motion of the figures. The program is written in Extended Basic and originally came from the Home Computer Magazine. I rewrote the program to display the various stages of the figure in motion and added more figures and a roadway for them to cartwheel on.



```
100 ! *******************
110 ! *  SPRITE DEMO 2   *
120 ! *******************
130 ! *99'ER VER 1.5.1XB*
140 ! * DEMO OF SPRITE  *
150 ! * ANIMATION USING *
160 ! * DATA STATEMENTS *
170 ! *MODIFIED BY 3 PARR
180 ! *  CC99'ERS 2/85  *
190 ! *******************
200 CALL CLEAR
210 CALL CHAR(60,"FFFF000FF0
OFOFFF"):: CALL CHAR(65,"FFF
FFFFFFFFFFFF"):: CALL COLOR
(5,3,3):: CALL COLOR(4,2,11)
220 CALL CHAR(55,"8046042528
700210"):: CALL COLOR(3,16,1
5)
230 DIM I$(17),C$(17)
240 GOSUB 360 !CASTER
250 FOR I=0 TO N :: CALL CHA
R(136-4*I,C$(I))
260 NEXT I
270 CALL SPRITE(#5,136,2,80,
10,#6,132,2,80,40,#7,128,2,8
0,70,#8,124,2,80,100,#9,120,
2,120,10)
280 CALL SPRITE(#10,116,2,12
0,40,#11,112,2,120,70,#12,10
8,2,120,100,#13,108,2,160,10
,#14,104,2,160,40)
290 CALL SPRITE(#15,100,2,16
0,70,#16,96,2,160,100)
300 CALL CLEAR
310 CALL HCHAR(9,1,60,32)::
CALL HCHAR(10,1,65,480):: CA
LL HCHAR(7,1,55,64)
320 CALL SPRITE(#1,136,2,34,
30,0,-6,#2,136,7,34,60,0,-6,
#3,136,13,30,90,0,-8,#4,136,
5,26,120,0,-20)
330 CALL MAGNIFY(4)
340 FOR I=0 TO N :: CALL PAT
TERN(#1,136-4*I,#2,136-4*I,#
3,136-4*I,#4,136-4*I):: GOSU
B 410 :: NEXT I :: GOTO 340
350 END
360 REM SUBROUTINE CASTER
370 READ NAM$,N
380 FOR I=0 TO N
390 READ I$(I),C$(I):: NEXT
I
400 RETURN
410 REM SUBROUTINE DELAY
420 FOR J=0 TO 6 :: NEXT J
430 RETURN
440 DATA MAN6N81,12
450 DATA MAN#1,00060909060F0
F0F1E060F0F190804080000000000
0000000000020508000000
460 DATA MAN#2,0304040307072
F130303070606020700008080008
090D0A08080808080808000
470 DATA MAN#2.5,00070903060
F0F172F0606060F090818000000
0000000000000000000804080
480 DATA MAN#3,00070903060F0
F172F0606060F09081800000000
00000000000000000804080
490 DATA MAN#4,000018241C0C1
C2C4E1606070602020600000000
00000000040A0000000000
500 DATA MAN#5,000000000000
0387FDE96624281000100000000
00020508000000000008000
510 DATA MAN#6,00061424140C0
C0C0C1C1E1E1E1D0C1000000000
000000000000000000008080
520 DATA MAN#6.5,0000002020 1
84C7C0C0C0E0606090E04000000
0000000000000000000000008080
530 DATA MAN#7,000000000000
04080402F1E376640C000000000
00000004080000000804020
540 DATA MAN#8,000000110A060
30101010303062A1206000000844
850A0C0C080800000000000
550 DATA MAN#1,00060909060F0
F0F1E060F0F1908040800000C
0000000000020508000000
560 DATA MAN#3,00070903060F0
F172F0606060F09081800000000
000000000000000000804080
570 DATA MAN#2.5,00070903060
F0F172F0606060F090818000000
0000000000000000000804080
```

TIPS FROM THE TIGERCUB

#36

Copyright 1986

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.

Some old business to take care of -

Tom Wible (? - handwritten signature), in the MANNERS NEWSLETTER for April, points out that I am all wrong in my comments about updating a FIXED SEQUENTIAL file. There is no such thing as a fixed sequential or fixed relative file, only fixed files accessed sequentially or randomly (relative). Sequential and relative are access modes, not file attributes. There is no reason to open a fixed file in anything other than RELATIVE mode, because if you do not specify the REC clause in your INPUT or PRINT, the computer defaults to sequential processing.

In one paragraph, that gentleman told me something about files I had't learned from the TI manuals and from the 2ØØØ+ newsletters on my shelf. File handling is apparently easy to understand for those who have had formal computer training, but it is a frustrating mystery to those of us who try to learn by hacking it. Won't somebody please write a series of articles, somewhere, in plain, non-computerese English?

And here is the last word on printing lines of more than 8Ø characters out of the TI-Writer Formatter, by W. Stewart Ash in a MANNERS newsletter of May-June 1986. It is only necessary to use the .FI command, and to set the right margin to the length you want, for example .FI;RM 12Ø for lines of 12Ø characters; and then use .TL or CTRL U commands to select a type font which will fit that many characters on a line (136 or 132 in condensed, depending on your printer; 96 in elite).

Here's a new way to make music, for you Basic-only users, music programmers and country music fans.

```
1ØØ CALL CLEAR
11Ø PRINT "          WILDWOOD FL
OWER": : :" on the hammered
 dulcimer": : : : : : : :"
          by Jim Peterson"
12Ø DIM S(26)
13Ø F=262
14Ø FOR N=1 TO 25
15Ø S(N)=INT(F*1.Ø59463Ø94^(
N-1))
16Ø NEXT N
17Ø READ N
18Ø C=S(N)
19Ø D=S(N)
2ØØ CALL SOUND(-35Ø,S(N),Ø)
21Ø RESTORE 35Ø
22Ø FOR J=1 TO 63
23Ø GOSUB 26Ø
24Ø NEXT J
25Ø GOTO 2ØØ
26Ø READ N
27Ø CALL SOUND(-35Ø,S(N),Ø)
28Ø X=1^1ØØ
29Ø CALL SOUND(-35Ø,S(N),Ø,C
,9)
3ØØ X=1^1ØØ
31Ø CALL SOUND(-35Ø,S(N),Ø,C
,9,D,19)
32Ø D=C
33Ø C=S(N)
34Ø RETURN
35Ø DATA 5,6,8,8,1Ø,13,5,5,6
,5,3,3,5,3,1,1
36Ø DATA 5,6,8,8,1Ø,13,5,5,6
,5,3,3,5,3,1,1
37Ø DATA 8,13,17,17,17,15,13
,13,8,8,1Ø,1Ø,13,1Ø,8,8
38Ø DATA 1,1,1,3,5,5,8,5,3,3
,5,3,1,1,1
```

Lines 12Ø-16Ø set up a scale of two octaves, beginning with the frequency in line 13Ø - to change the key, just change that frequency. Lines 17Ø-19Ø set up the initial values, line 2ØØ prevents a pause while data is being restored. Then the routine reads the data and plays the music.

Note the dummy calculation in lines 28Ø and 3ØØ, which does nothing but create a brief pause while the value of X is computed. This is a good method for a delay because it can be adjusted so exactly by changing the exponent, but use a value of 1 to avoid a numeric overflow.

To write your own music by this method, just list the notes of a 2-octave scale from your starting frequency C C# D Ef E F F# 6 - etc. and number them 1 to 25.

Then, list the notes of your song by their number in the DATA statements. For a longer note, list it twice or more. Change the TO value in line 22Ø to your total number of notes, and RUN!

Here's one just to doodle around with. You can create a 3-dimensional maze, save it to tape or disk, or erase it and watch the computer draw it again.

```
1ØØ CALL CLEAR :: CALL CHAR(
128,"FFØØØØØØØØØØØØØØFF81818
181818181818Ø8Ø8Ø8Ø8Ø8ØFFFF8
Ø8Ø8Ø8Ø8Ø8Ø81")
11Ø CALL CHAR(132,"81Ø1Ø1Ø1Ø
1Ø1Ø1FFFFØ1Ø1Ø1Ø1Ø1Ø18181ØØØ
ØØØØØØØØØ81"):: T=1 :: DIM K
$(15)
12Ø DISPLAY AT(3,7):"GORDIAN
 KNOT": : :TAB(12);"by Jim P
eterson"
13Ø DISPLAY AT(9,1):" Use ar
row keys to create a":"3-dim
ensional maze."
14Ø DISPLAY AT(12,1):" You m
ay at any time press":"Q to
clear the screen, or P":"to
save a manually created":"sc
reen."
15Ø T=1 :: DISPLAY AT(17,1):
"Choose - ":" (1) Manual":"
(2) Automatic":" (3) Retrace
":" (4) Load"
16Ø ACCEPT AT(17,11)VALIDATE
("1234")SIZE(1)BEEP:Q :: ON
Q GOTO 17Ø,22Ø,29Ø,4ØØ
17Ø GOSUB 44Ø
18Ø CALL KEY(3,K,ST):: IF ST
=Ø THEN 18Ø ELSE D=POS("EDXS
QP",CHR$(K),1)+1 :: ON D GOT
O 18Ø,2ØØ,2ØØ,2ØØ,2ØØ,19Ø,36
Ø
19Ø CALL CLEAR :: GOTO 15Ø
2ØØ D=D-1 :: IF ABS(D-D2)=2
THEN 18Ø :: GOSUB 51Ø :: IF
D<>D2 THEN GOSUB 45Ø
21Ø GOSUB 49Ø :: GOSUB 52Ø :
: GOTO 18Ø
22Ø GOSUB 44Ø :: RANDOMIZE
23Ø D=D+1+(D=4)*4 :: CALL KE
Y(Ø,K,ST):: IF ST=Ø THEN 25Ø
24Ø IF K=8Ø THEN 36Ø ELSE IF
 K=81 THEN CALL CLEAR :: GOT
O 15Ø
25Ø T=INT(4*RND+2)*2-INT(2*R
ND)
26Ø FOR J=1 TO T :: IF D<>D2
 THEN GOSUB 45Ø
27Ø GOSUB 49Ø :: CH=128-(D=1
)-(D=3):: CALL GCHAR(R,C,G):
: IF G<>32 THEN IF INT(2*RND
+1)<>1 THEN CH=G
28Ø GOSUB 53Ø :: NEXT J :: G
OTO 23Ø
29Ø IF LEN(K$(1))=Ø THEN DIS
PLAY AT(24,1):"CAN'T DO THAT
" :: GOTO 16Ø
3ØØ CALL CLEAR :: GOSUB 44Ø
:: FOR J=1 TO T :: FOR JJ=1
TO LEN(K$(T)):: D=POS("EDXS"
,SEG$(K$(T),JJ,1),1)
31Ø IF D=Ø THEN 35Ø :: IF D<
>D2 THEN GOSUB 45Ø
32Ø GOSUB 49Ø :: CH=128-(D=1
)-(D=3):: CALL GCHAR(R,C,G):
: IF G=32 THEN GOSUB 53Ø ::
GOTO 35Ø
33Ø K=ASC(SEG$(K$(T),JJ+1,1)
):: IF K<>67 AND K<>79 AND K
<>85 THEN JJ=JJ+1 :: GOTO 33
Ø
34Ø GOSUB 48Ø :: GOSUB 53Ø
35Ø NEXT JJ :: NEXT J :: GOT
O 16Ø
36Ø IF LEN(K$(1))>Ø THEN 37Ø
 :: DISPLAY AT(12,1)ERASE AL
L:"CAN'T DO THAT!" :: GOTO 1
5Ø
37Ø DISPLAY AT(12,1)ERASE AL
L:"Save to - ":" (1)Cassette
":" (2)Disk" :: ACCEPT AT(12
,11)VALIDATE("12")SIZE(1):S
:: IF S=1 THEN OPEN #1:"CS1"
,INTERNAL,OUTPUT,FIXED 192 :
: GOTO 39Ø
38Ø DISPLAY AT(16,1):"Filena
me DSK" :: ACCEPT AT(16,13):
F$ :: OPEN #1:"DSK"&F$,INTER
NAL,FIXED 192,OUTPUT
39Ø PRINT #1:T :: FOR J=1 TO
 T :: PRINT #1:K$(J):: K$(J)
="" :: NEXT J :: CLOSE #1 ::
 GOTO 15Ø
4ØØ DISPLAY AT(12,1)ERASE AL
L:"Load from -":" (1)Cassett
e":" (2)Disk" :: ACCEPT AT(1
2,13)VALIDATE("12")SIZE(1)BE
EP:L :: IF L=1 THEN OPEN #1:
"CS1",INTERNAL,FIXED 192,INP
UT :: GOTO 42Ø
41Ø DISPLAY AT(16,1):"Filena
me? DSK" :: ACCEPT AT(16,14)
BEEP:F$ :: OPEN #1:"DSK"&F$,
INTERNAL,FIXED 192,INPUT
42Ø INPUT #1:T :: FOR J=1 TO
 T :: INPUT #1:K$(J):: NEXT
J :: CLOSE #1 :: GOTO 3ØØ
43Ø CLOSE #1 :: GOTO 3ØØ
44Ø CALL CLEAR :: CALL COLOR
(13,5,11):: R,R2=12 :: C,C2=
14 :: D2=3 :: CH=129 :: CALL
 HCHAR(R2,C2,CH):: RETURN
45Ø CH2=128+((D2=1)*(D=2)*3)
+((D2=1)*(D=4)*5)+((D2=3)*(D
=2)*2)+((D2=3)*(D=4)*4)+((D2
=2)*(D=1)*4)+((D2=2)*(D=3)*5
)
46Ø CH2=CH2+((D2=4)*(D=1)*2)
+((D2=4)*(D=3)*3):: CALL HCH
AR(R2,C2,CH2):: RETURN
47Ø CALL KEY(3,K,ST):: IF ST
=Ø THEN 47Ø ELSE IF POS("COU
",CHR$(K),1)=Ø THEN 47Ø
48Ø GOSUB 51Ø :: IF K=67 THE
N CH=134 :: RETURN ELSE IF K
=85 THEN CH=G :: RETURN ELSE
 RETURN
49Ø R=R+(D=1)-(D=3):: IF R<3
 OR R>24 THEN R=R2
5ØØ C=C+(D=4)-(D=2):: IF C<3
 OR C>3Ø THEN C=C2 :: RETURN
 ELSE RETURN
51Ø IF Q<>1 THEN RETURN ELSE
 K$(T)=K$(T)&CHR$(K):: IF LE
N(K$(T))<193 THEN RETURN ELS
E T=T+1 :: RETURN
52Ø CH=128-(D=1)-(D=3):: CAL
L GCHAR(R,C,G):: IF G<>32 TH
EN GOSUB 47Ø
53Ø CALL HCHAR(R,C,CH):: R2=
R :: C2=C :: D2=D :: RETURN
```

I think that educational programs should teach, not just test. This one makes up the kind of problems we all hated in school, but if you get the answer wrong it will show you how to work it.

```
1ØØ CALL CLEAR :: RANDOMIZE
11Ø DATA LUMBERJACK,CUT,CORD
S OF WOOD,BOY,PICK,QUARTS OF
 BERRIES,ELEPHANT,EAT,BALES
OF HAY,COW,GIVE,GALLONS OF M
ILK
12Ø FOR J=1 TO 4 :: FOR L=1
TO 3 :: READ M$(J,L):: NEXT
L :: NEXT J
13Ø A=INT(5*RND+2):: IF A=A2
 THEN 13Ø ELSE A2=A
14Ø B=INT(9*RND+2):: IF B=B2
 THEN 14Ø ELSE B2=B
15Ø C=INT(9*RND+2):: IF C=C2
 THEN 15Ø ELSE C2=C
155 X=B/C/A :: IF LEN(STR$(X
))>4 THEN 13Ø
16Ø D=INT(4*RND+1):: IF D=D2
 THEN 16Ø ELSE D2=D
17Ø DISPLAY AT(3,1)ERASE ALL
:"IF";A;M$(D,1);"S CAN ";M$(
D,2):B;M$(D,3);" IN":C;"DAYS
,"
18Ø DISPLAY AT(6,1):"HOW MAN
Y ";M$(D,3);" CAN 1 ";M$(D,1
);" ";M$(D,2);" IN 1 DAY?"
19Ø ACCEPT AT(9,1)VALIDATE(N
UMERIC)BEEP:Q
2ØØ IF Q<>X THEN 3ØØ :: DISP
LAY AT(11,1):"CORRECT!"
21Ø DISPLAY AT(23,1):"PRESS
ANY KEY" :: CALL KEY(Ø,K,ST)
:: IF ST=Ø THEN 21Ø ELSE 13Ø
3ØØ DISPLAY AT(11,1):"NO -":
"IF";A;M$(D,1);"S CAN ";M$(D
,2):B;M$(D,3);" IN":C;"DAYS,
"
31Ø DISPLAY AT(15,1):"THEN";
A;M$(D,1);"S CAN ";M$(D,2):B
;"/";C;M$(D,3);" IN 1 DAY":B
;"/";C;"=";B/C
32Ø DISPLAY AT(19,1):"SO 1 "
;M$(D,1);" CAN ";M$(D,2);B/C
;"/";A;M$(D,3);" IN 1 DAY":B
/C;"/";A;"=";X :: GOTO 21Ø
```

Here's a new way to put a title on the screen -

```
1ØØ !SCATTERPRINT by Jim Pet
erson
11Ø CALL CLEAR :: M$="TIGERC
UB SOFTWARE" :: L=LEN(M$):
IF L>28 THEN 11Ø :: C$=SEG$(
"ABCDEFGHIJKLMNOPQRSTUVWXYZ[
\",1,L)
12Ø FOR J=1 TO L :: RANDOMIZ
E :: X=INT(LEN(C$)*RND+1)::
Y=ASC(SEG$(C$,X,1))-64
13Ø DISPLAY AT(2,13-L/2+Y):S
EG$(M$,Y,1);:: C$=SEG$(C$,1,
X-1)&SEG$(C$,X+1,255):: NEXT
 J
14Ø GOTO 14Ø
```

This one is very basic, but if you have Terminal Emulator II, Speech Synthesizer, and a preschool child, it's a fine way to learn the alphabet, the keyboard, to spell his name, or just to have fun with - try a string of KK's for a train chugging uphill.

```
1ØØ OPEN #1:"SPEECH",OUTPUT
11Ø CALL KEY(3,K,S)
12Ø INPUT M$
13Ø PRINT #1:M$
14Ø GOTO 12Ø
```

Memory full - Jim P.

VDP UTILITY---

John Behnke

eprinted from the Chicago Times, the newsletter of the Chicago Users Group

If you can not run a BASIC program in X/Basic because of CALL COLOR, or CALL CHAR statements which are above the limits of X/Basic, this program will allow you now to do so. To use, simply enter the program and save it off as a MERGE file (SAVE DSK1.VDPUTIL, MERGE). The rest of the instructions can be found in the programs remarks.

```
1 CALL VDPUTIL2
32700 !"VDP UTILITY II"
32701 !BY JOHN BEHNKE
32702 !   5755 W. GRACE
32703 !   CHICAGO IL 60634
32704 !WILL CONVERT ANY BASI
C
32705 !PROGRAM TO X-BASIC
32706 !DIRECTIONS: LOAD BASI
C
32707 !PROGRAM INTO X-BASIC.
32708 !THEN INPUT:
32709 ! "MERGE DSK1.VDPUTIL2
"
32710 !WHEN FINISHED, RE-SAV
E
32711 !GAME. THE RESULTING
32712 !PROGRAM WILL RUN IN
32713 !X-BASIC.
32714 SUB VDPUTIL2
32715 CALL CLEAR :: CALL INI
T :: CALL LOAD[8196,63,232]
32716 CALL LOAD[16360,80,79,
75,69,82,32,38,12,80,79,75,6
9,86,32,37,164,80,69,69,
75,86,32,37,36]
32717 CALL LOAD[9491,100]
32718 CALL LOAD[9508,2,224,3
7,20,3,0,0,0,2,0,0,100,200,0
,37,18,4,192,2,1,0,1,4,3
2,32,12,4,32]
32719 CALL LOAD[9536,32,24,1
8,184,192,32,131,74,2,1,37,0
,208,160,131,18,9,130,2,
34,255,255,4,32,32,44]
32720 CALL LOAD[9562,4,197,2
09,34,36,255,9,132,19,21,4,1
95,60,224,37,18,200,5,13
1,76,200,5,131,78,200,5]
32721 CALL LOAD[9588,131,80,
2,5,64,0,161,68,2,131,0,1,17
,6,2,5,65,0,161,67,6,196
,200,4,131,76]
32722 CALL LOAD[9614,200,5,1
31,74,4,192,192,66,5,129,4,3
7,254]
32723 CALL LOAD[9636,2,224,3
7,20,3,0,0,0,4,192,2,1,0,1,2
00,1,37,18,4,32,32,12,4,
32,32,24,18,184]
32724 CALL LOAD[9664,200,32,
131,74,37,0,184,32,131,18,37
,19,2,3,0,2]
32725 CALL LOAD[9680,4,192,1
92,67,4,32,32,12,4,32,32,24,
18,184,216,224,131,75,37
,0,5,131,136,3]
32726 CALL LOAD[9704,37,18,2
2,242,192,32,37,0,2,1,37,2,1
92,131,2,34,255,254,4,32
,32,36]
32727 CALL LOAD[9726,4,192,2
16,0,131,124,2,224,131,224,4
,96,0,112]
32728 CALL LOAD[9740,3,0,0,0
,4,192,2,1,0,1,4,32,32,12,20
0,32,131,74,37,18,2,1,0,
2,4,32,32,12,4,32]
32729 CALL LOAD[9770,32,24,1
8,184,192,32,131,74,208,32,3
7,19,4,32,32,48,4,91]
32730 CALL LOAD[8194,39,04]
32731 SUBEND
32732 SUB CHAR[A,A$]:: L=LEN
[A$]
32733 A$=A$&RPT$["0",16-L]
32734 FOR I=1 TO 16 STEP 2
32735 A1$=SEG$[A$,I,1]
32736 A2$=SEG$[A$,I+1,1]
32737 IF A1$<":" THEN A1=VAL
[A1$]*16 ELSE A1=[ASC[A1$]-5
5]*16
32738 IF A2$<":" THEN A1=A1+
VAL[A2$]ELSE A1=A1+ASC[A2$]-
55
32740 CALL LINK["POKEV",767+
8*A+[I+1]/2,A1]
32741 NEXT I
32742 SUBEND
32743 SUB COLOR[A,B,C]
32744 CALL LINK["POKEV",2063
+A,[B-1]*16+C-1]
32745 SUBEND
```

CEDAR VALLEY 99'ER U.G.
288 WINDSOR DR. NE
CEDAR RAPIDS, IA 52402

DECATUR 99'ER HOME COMPUTER U.G.
P.O. BOX 726
DECATUR, ILLINOIS 62525

NEXT MEETING DATE:
THURSDAY, JULY 10, 1986

ALL MEETING DATES:
6:30 PM TO 8:30 PM
DECATUR PUBLIC LIBRARY
SECOND FLOOR BOARD ROOM

>>>>JULY MEETING DATE<<<<<

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

DECATUR 99er HOME COMPUTER USERS GRP
APPLICATION FOR MEMBERSHIP

DATE / /85

NAME______________________________

ADDRESS___________________________

CITY____________________ZIP______

PHONE_____________________________

WORK PHONE________________________

DUES: MEM,STUDENT $15
ADD'L FAMLY $ 5
OR NEWSLETTER ONLY $12 $____
(25 MAX)