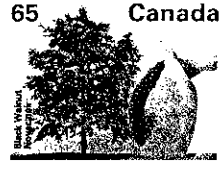
Newsletter Nine-T-Nine

DECEMBER 1992 ISSUE

Texas Instruments
USERS GROUP
TORONTO

According to Bart Simpson, "If TV has taught me anything, it's that miracles always happen to poor kids at Christmas. It happened to Tiny Tim, it happened to the Smurfs, and it's going to happen to us." Will the 9T9ers get a new editor in '93?



FROM:
9T9 USERS GROUP
15 KERSDALE AVE.
TORONTO, ONT., M6M-1C9
CANADA

NEWSLETTER NINE-T-NINE

9T9 Users Group

Merry Christmas

☆

9T9 USERS GROUP EXECUTIVE COMMITTEE

PRESIDENT Steve Mickelson (657-1494)
VICE-PRESIDENT Neil Allen (236-0842)
SECRETARY/MEMBERSHIPS Randy Rossetto (469-3468)
TREASURER/OFFICER AT LARGE Cecil Chin (671-2052)

LIBRARY DIRECTORS

Gary Bowser (960-0925)
Andy Parkinson (275-4427)
Steve Findlay (416) 727-6807

NEWSLETTER EDITOR

Steve Mickelson (657-1494)

MEMBERSHIP FEE'S

FULL MEMBERSHIP..... $30.00 / year
NEWSLETTER SUBSCRIPTION ..... $20.00 / year

All memberships are household memberships. A newsletter subscription is only for those who do not wish to attend meeting, but wish to receive our newsletter and have access to our library. You are welcome to visit one of our general meetings before joining the group. If you wish more information contact either our president, in writing, at the club address on the front cover or by phone.

The meetings are usually held on the last Wednesday of each month,(exceptions are December's meeting date, usually mid-month and the months of July and August, when there are no meetings. Consult this issue of Newsletter 9T9 for the date and time of the next meeting. Meetings are usually held at Neil Allen's place, 52 Graystone Gardens,south of Bloor St., just west of Islington Ave., at. 7:30 P.M. from 7:30 - 10:30 PM.

BBS

The 9T9 Users Group supports the Toronto BBS, The TI Tower BBS #(416) 921-2731, 300/1200/2400 BPS, 24 hrs. Sysop, Gary Bowser.

MAILING ADDRESS:

9T9 Users Group, 15 Kersdale Ave., Toronto, Ontario, M6M 1C9, Canada

COMMERCIAL ADVERTISING

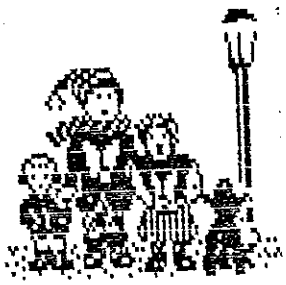
Any business wishing to reach our membership may advertise in our newsletter.
The rates are as follows. (width by height):
FULL PAGE (7" x 10") $30.00
HALF PAGE (7" x 5") $15.00
QUARTER PAGE (7" x 2 1/2") $ 7.50
Please have your ad's camera ready and paid for in advance. For more information contact the editor. Don't forget, that any member wishing to place ad's, may do so free of charge as long as they are not involved in a commercial enterprise.

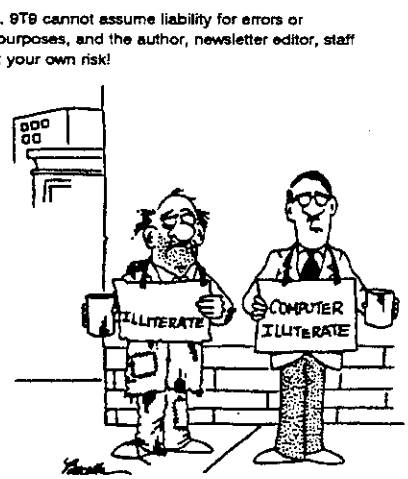
NEWSLETTER ARTICLES

Members are encouraged to contribute to the newsletter in the form of articles, mini programs, helpful tips, hardware modifications, jokes, cartoons and questions. Any article may be submitted in any form by mail or modem. We welcome the reprinting of any article appearing in this newsletter providing credit is given to the author and 9T9. If more information is required, call the editor. The names, 9T9, Nine-T-Nine, Newsletter 9T9, 9T9 Users Group, and Nine-T-Nine Users Group are Copyright, (c) ,1982-1992, by the 9T9 Users Group of Toronto, Canada, all rights reserved.

DISCLAIMER

Opinions expressed in this newsletter are those of the writers and are not necessarily those of the 9T9 USERS' GROUP. 9T9 cannot assume liability for errors or omissions in articles, programs or advertisements. Any hardware modification or project is presented for informational purposes, and the author, newsletter editor, staff and/or 9T9 Users Group &cannot be held liable for any damage to the user's equipment. All such projects are done at your own risk!





MEMBERSHIP RENEWALS & ELECTIONS NEXT MONTH!

TIDBITS #64

-By Steve Mickelson, President 9T9 Users Group
Compuserve 76545,1255; Delphi SMICKELSON; GEnie S.MICKELSON

That's all folks:
This edition will mark my last TIdbits. Rather than reflect on the last six years of news and views in past editions of TIdbits, I prefer to reprint an article sent to me, from Bill Gard, of the Ottawa group. Bill, also states in a letter that the Ottawa TI group plans to host another TI Fest, in April 1993.

I would like to thank the support from all our readers and trust that you give support to the next newsletter editor. It's been a great six year's plus and I hope the Nine-T-Nine group and newsletter continue, for at least as long! Don't forget membership renewals and elections, in January 1993!

On Proof-Reading

by W. Leonard Tabbs

As I struggle to follow the late Ida McCargar as Newsletter Librarian, I have the priviledge of reading or scanning newsletters from other TI User Groups across our country and Canada. The number of editorial errors is fairly substantial, all newsletters being viewed as a whole. Before starting to complain, one must consider that the world's TI Newsletter editors are among the unsung heroes of our remaining TI world!

Ideally when anyone has turned in an article to the newsletter editor, it would be nice for that author to have the opportunity to proof-read the article before it "goes to press". This, though ideal, simply proves to be impractical logistically. As anyone involved with producing newsletters for our TI friends well knows, a tremendous amount of work is thrust upon the editor (and unfortunately--more often than not--at the last minute!). Some of the newsletters from around the country I have seen are pretty clearly the sweat and toil of one person alone!

Think about the work your Newsletter Editor does. How many of us take for granted what they do? When they ask for help, how many in your group volunteer? How many writers understand the editorial pressures created by lackadaisical fumbling on getting articles in to the editor on time? If writers are not good spellers, how many take the time to get a spouse or friend to check the article for grammatical or spelling errors so your Newsletter Editor will not be stuck with this extra chore(!)? Remember, too, newsletter editors have families, jobs, and other community obligations besides their editorial duties! And yet, many of them find the time to do beautiful work dressing up their newsletters with really neat graphics as well!

So the next time you wince at errors in your newsletter, remember to be thankful you have a newsletter at all and that you are lucky enough to have a person dedicated enough to serve as editor. After all, not only has the editor given all those countless hours to produce the letter but probably has also (often single-handedly!) had to go out to have it copied and then spends more hours collating the pages, stapling them, folding them, sticking on address labels, and putting on stamps (not many groups have large enough membership to use metered mail). And who takes them to the post office?... These editors have to rank very close to Saints!

From the advantage I have of seeing so many different newsletters from the TI world, I can tell you our own SW99er newsletter certainly appears to be among the best. This is not a boast but a thank-you to our own editor! (Personally, I am glad to see newsletters regardless of how well they are put together. I would rather see a newsletter full of typos (though hopefully not too many in program listings!) rather than no newsletter at all)! As many newsletters have folded in recent years--sadly including some outstanding ones--we (the TI newsletter world) are unmistakably a shrinking world. In many cases of the surviving newsletters, it appears that only a loyal few people in each group keep the current newsletters coming. Let us give thanks and doff our hats to all of them!!

SouthWest Ninety-Niners/Oct '92

->LAST ISSUE<-
->RENEW NOW!<-

FAST EXTENDED BASIC

by Lucie Dorais

This year, I'll not just give you one Christmas song, but a collection of 20 short tunes easily played from a Jukebox. The DATA lines for the songs seem like a big mountain until you know how to analyze them, which I will tell you.

This program is not entirely mine: the DATA from the Christmas songs, and more importantly the algorithm to play them, comes straight from Texas Instruments' program MYSTERY MELODY (copyright 1980, TI-Basic), which contains short quotes from 50 songs and a quiz. There is a feature to listen to each song individually if you want to learn them, but it is complicated, and TI-Basic is slow. So I "cannibalized" the DATA statements and the play routine, transformed them into XB, and added a three-screen menu. Since each screen had twenty titles, I added 10 "classical" pieces to round the number to 60. Finally, because the program took some time to analyze each data statement before playing it, I added some animation: when you chose a title, a little jukebox machine appears at the bottom, and the disk is moved to the turntable. Then, when the music plays, the screen changes colour with each note while musical notes move from right to left.

When the program was finally ready for publishing in this Newsletter (back in 1989), I had second thoughts: too many DATA statements, and the copyright problem... so I put it aside. With Christmas 92 around the corner, I dug it up, reduced the number of songs to 20, and changed the colours to look more like Christmas. Since there were only 14 Christmas songs in the TI program, I added six of my "classical" additions (lines 690-700, 720, 640, 790 and 820).

```
100 ! ** XMASBOX ** Xmas songs & play routine: TI's MYSTERY
    MELODY game; the rest by  L.Dorais/Ottawa U.G. June 89, Xmas
    mods Nov. 92
110 DATA DUMMY
120 DIM H(25),S(60),D(60),PS(48),T$(20) :: RANDOMIZE :: READ C$
130 CALL CLEAR :: CALL SCREEN(16) :: CALL CHAR(128,"000000FF
    FF",129,"00E0F0F8FCF8F0E0",136,"0",130,"04060504
    74FCFC70")
140 CALL CHAR(123,"005555555555FF7F005454545454FEFC000
    02020A8A82020",91,"FFFFFFFFFFFF",92,"00000000007F7
    F000000000000FEFE")
150 FOR I=1 TO 12 :: CALL COLOR(I,13,1) :: NEXT I :: CALL
    COLOR(13,9,1) :: CALL MAGNIFY(2)
160 B$=CHR$(130) :: C$="  7     " :: SC$=C$&RPT$(B$,11) ::
    A$=C$&B$&C$&" 2 "&B$
170 GOTO 190 :: A,B,D1,H1,I,K,KEY,N,O1,R,SC,ST,T,X,Z :: CALL
    SPRITE :: CALL MOTION :: CALL POSITION :: CALL DELSPRITE
180 CALL SOUND :: CALL KEY :: CALL HCHAR :: CALL SCREEN :: !@P-
190 DISPLAY AT(2,1):SC$:A$:C$&B$&" JUKEBOX "&B$:A$:SC$
200 DISPLAY AT(11,6):"XMAS SONGS FROM": :" 3  TI's MYSTERY
    MELODY"::TAB(10);"& MORE..." :: CALL HCHAR(19,3,128,28)
210 CALL SPRITE(#20,125,13,177,56,-2,0,#21,123,13,177,26,
    #22,123,13,177,38,#23,,124,13,177,54)! move disk up/record
    rack
220 FOR X=1 TO 10 :: A=INT(144*RND)+1 :: B=-(INT(30*RND)+20
    :: CALL SPRITE(#X,130,1,A,255,0,B) :: NEXT X ! invisible notes
230 T=200 :: H1=2^(1/12) :: CALL MOTION(#20,0,10)! tempo, scale,
    move dsk rght
240 H(0)=30000 :: H(1)=262 :: FOR X=2 TO 25 :: H(X)=H(X-1)*H1::
    NEXT X
250 A$="RC1D1EF1G1A1B%.#84+2-111" :: FOR X=1 TO LEN(A$)::
    Z=ASC(SEG$(A$,X,1)) :: PS(Z-34)=X :: NEXT X ! fill pointer array
260 FOR I=1 TO 20 :: READ T$(I),SC$ :: NEXT I ! title array
270 ! = = = = = display song menu = = =-= =
280 CALL DELSPRITE(#20) :: CALL COLOR(#21,1,#22,1,#23,1)::CALL
    CLEAR 290 CALL HCHAR(21,1,128,32) :: R=1
```

```
300 FOR X=1 TO 20 :: DISPLAY AT(X,3):T$(X) :: NEXT X :: CALL
    HCHAR(R,4,129)
310 DISPLAY AT(22,1):" ² SELECT SONG with EX keys,":" ² press
    <ENTER> to hear it,":TAB(9);"<Q> to quit"
320 CALL KEY(0,KEY,ST) :: IF ST=0 THEN 320 :: IF KEY=81 THEN
    END ELSE IF KEY=13 THEN 380
330 IF KEY<>69 AND KEY<>88 THEN 320 ELSE CALL HCHAR(R,4,
    32) ! erase cursor
340 IF KEY=69 THEN R=R-1 :: IF R=0 THEN R=20 :: GOTO 360 !
    cursor up
350 IF KEY=88 THEN R=R+1 :: IF R=21 THEN R=1 ! cursor down
360 CALL HCHAR(R,4,129) :: GOTO 320 ! move cursor
370 ! ===== analyze a song =====
380 DISPLAY AT(22,1):"":"":"" :: CALL SOUND(50,2000,0):: CALL
    SOUND(50,1500,0)
390 CALL COLOR(#21,13,#22,13,#23,13) :: DISPLAY AT(24,26):"[[" !
    rack/turntabl
400 CALL SPRITE(#20,125,13,177,4*INT(10*RND+6),-10,0) ! move
    disk up
410 RESTORE 670 :: CALL MOTION(#20,0,3) ! move disk right
420 READ C$,SC$ :: IF C$<>T$(R)THEN 420 ! find song
430 Z=LEN(SC$) :: A=9-INT(Z-23)/8 :: IF A>3 THEN CALL
    MOTION(#20,0,A) ! adjust disk motion
440 O1=0 :: D1=2 :: K=0 :: FOR I=1 TO Z ! def. octave,duration,start
    counter
450 C$=SEG$(SC$,I,1) :: X=PS(ASC(C$)-34) :: IF X=0 THEN 520 !
    char. pointer
460 IF X<=13 THEN K=K+1 :: S(K)=X-1-O1*(X>1) :: D(K)=D1 ::
    GOTO 520 ! note pointer for H array
470 ON X-13 GOTO 480,490,480,500,500,510,500,510,520,520,500
480 IF S(K)=0 THEN 520 ELSE S(K)=S(K)+X-15 :: GOTO 520 !
    sharp/flat
490 D1=1.5*D1 :: D(K)=D1 :: GOTO 520 ! dotted note
500 D1=X-16 :: D(K)=D1 :: GOTO 520 ! duration (1,2,4,8)
510 O1=-12*(X=19)! octave
520 CALL POSITION(#20,A,B) :: IF B<218 THEN 530 ELSE CALL
    MOTION(#20,0,0)
530 NEXT I
540 CALL POSITION(#20,A,B) :: IF B<218 THEN 540 ELSE CALL
    MOTION(#20,4,0)
550 CALL POSITION(#20,A,B) :: IF A<170 THEN 550
560 CALL DELSPRITE(#20) :: DISPLAY AT(23,26):"\]" ! disk on turntable
570 ! ===== play the song =====
580 CALL HCHAR(1,1,136,640)! background
590 Z=14-LEN(T$(R))/2 :: DISPLAY AT(22,1):RPT$(" ",Z)&T$(R)! center
    title
600 DISPLAY AT(24,8)SIZE(-17):"(any key to stop)"
610 FOR X=1 TO 10 :: CALL COLOR(#X,INT(15*RND+2)) :: NEXT X !
    colour notes
620 FOR X=1 TO K :: CALL KEY(0,KEY,ST) :: IF KEY>-1 THEN 650
630 CALL SOUND(T*D(X),H(S(X)),1) :: CALL COLOR(14,1,INT(11*
    RND)+3) :: NEXT X ! play a note, colour background
640 FOR X=1 TO 270 :: NEXT X :: GOTO 620 ! delay, replay
650 FOR X=1 TO 10 :: CALL COLOR(#X,1) :: NEXT X :: CALL
    COLOR(14,1,16) :: GOTO 280 ! back to menu
660 ! ===== song data =====
670 DATA AULD LANG SYNE,+A4G.E8E4CD.C8D4AG.E8E4GA1A4G.
    E8E4CD.C8D4E8D8C2-A8A4G4+C1
680 DATA AWAY IN A MANGER,CB#.B%8A4AGFFEDC2C4C.D8C4
    CGEDCFA2
690 DATA BACH MINUET,+D-G8AB+CD4-GG+EC8DEF#G4-GG+
    CD8C-BAB4B#8BAGA4B8AGF#G2.
700 DATA BRAHMS' LULLABY,E8EG2E8EG2E8GB#4B.A8A4GD8EF
    2D8EF2D8FBAG4BB#2
710 DATA COME ALL YE FAITHFUL,GG2D4GA2DB4ABB#B2A4GG2F#4
    EF#GABF#2E4.D8D1
720 DATA COUNTRY GARDENS,B#B#8BA4AGG8FE4E8FG4CDFE.D8C
    2R8C4FFAG8AGFE4E8FG4CDFE.D8C2
730 DATA DECK THE HALLS,A.G8F#4EDEF#DE8F#GEF#4.E8D4C#D2
740 DATA ELVIRA (MOZART),F2.C4A2.F4B#1B%4AGFF#2GG1CE2GB
    %1+D4C-B%G#2AA1B#A2.A4B#2.B%4B%1+D-B2.B4B#1
750 DATA GREENSLEEVES,A+C2D4E.F8E4D2-B4G.A8B4B#2A4
    A.G#8A4B2.E
760 DATA "HARK, THE HERALD ANGELS",DGG.F#8G4BBA+DDD.C8-
    B4AB2D4GG.F#8G4BBA+D-AAGF#ED2
770 DATA JINGLE BELLS.E8EEREEEREGCDE2F8FFFFEEEEDDED4G
```

"Basic, my dear Watson. Basic!"

```
780 DATA JOY TO THE WORLD,B#B8AG4.F8E4DC.G8A4.A8B4.B8B#1
790 DATA MEMORY,+F.F2E8FGFDF4.F2E8FGFCD4.D2-B%8+CDC-B%
    A1.A4+C8C4.-GA8B%4+C8D4E8FEDC1.-A8FB#1D8D4E8F1
800 DATA O CHRISTMAS TREE,D.G4G8G4.AB4B8B2.A4B8B#4.F#
    A4G8G2.
810 DATA REST YE MERRY GENTLEMEN,DDAAGFEDCDEFGA2.D4DAA
    GFEDCDEFGA2.
820 DATA SCARBOROUGH FAIR,D2D4AGAE.F8E4D2.A+C4D2C4-A
    BGA2A4+D2D4C2-A4AGFDCDD2A4G2F4EDCD2.
830 DATA SILENT NIGHT,G4.A8G4E2.G4.A8G4E2.+D2D
    4C%2.C2C4-G2.
840 DATA THE FIRST NOEL,E8DC4.D8EFG2A8BB#4BAG2A8BB#
    4BAGABB#GFE2

850 DATA TOWN OF BETHLEHEM,AAAA%AB#B%DGFE8FG4CA2.A
    4AA+DCC-B%DGFE8FA4GF2.
860 DATA WE THREE KINGS,A2G4F2D4EFED2.A2G4F2D4EFED2.F
    2F4G2G4A2A4B#B%AGAGF2E4D1.
```

Editor's note: The superscript numbers printed between quotes (eg. " ^{5} ") indicates the number of spaces to be entered.

The DATA DUMMY in line 110 is just that... because one DATA statement has to precede the prescan (it is READ in line 120). The arrays are used as follows: the H array holds the frequency for each notes from middle C to C two octaves higher, and D will hold the Duration of each note of the song after it has been analyzed. PS will be filled by a pointer value according to each character used in the song strings (thus it is not entirely filled: see line 250 and its explanation below); this value will be used by Tex to decide what kind of character it has encountered: a note, a duration attribute, an octave, etc.; T$ will hold the 20 titles in the menu.

The program title screen is initialized and displayed by lines 1590- 200 (with the prescan in between), then the sprites that define the record rack are put at the lower left of the screen. In line 220, Tex places the random sprites for the musical notes, with random motion from right to left (thus the negative value of B); they are transparent at this stage, but they will be there so the program don't waste time placing them on screen each time a new song starts; this line also moves the disk (sprite #20) towards the right until the program is ready to display the song menu.

Lines 240-250 are straight from the TI program; they initialize the H array with all the frequencies: H(0) is the Rest, H(1) middle C, and the PS array that will hold a value pointer for each character used in the song strings (for example, PS(1) is for character 35, "#" to indicate a sharp, and holds the pointer 16 (it is the 16th char. in the A$ string); when Tex will analyze the song, in line 470, the value of 16-13=3 will send the program to line 480, which deals with sharps and flats); for that reason, do not alter the string A$ in line 250.

Line 260 READs all the song DATA and keeps the titles in the T$ array. The disk rack is made invisible (line 280, colour 1 is transparent), the moving disk disappears, the main menu is displayed (lines 280-310) and Tex asks you to press a key. There is a cursor to the left; you move it with the E or X keys (alpha lock down, no need for FCTN), and you press <ENTER> to pick a song (lines 320-360). After you press <ENTER>, you GOTO 380 to analyze the song.

Two notes confirm your choice, the bottom information is erased, and the jukebox's disk rack is again displayed (by colouring it green) at the lower left; the turntable is displayed at the bottom right, and the disk starts to move; a random column is chosen, since you do not always play the same song! The negative vertical value makes the disk go up. The CALL MOTION in line 410 has two values: 0 for the vertical motion makes the disk stop in its ascension, and the 3 value for the horizontal motion makes it move slowly towards the right. In line 430, after Tex knows the length of the song

string SC$, the disk motion is adjusted so that it does not pass by the turntable.

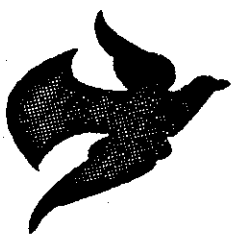
While the disk moves, the program continues: in order to find the correct DATA line for your song, Tex has to read all of them sequentially until it finds the correct title by comparing each DATA title, C$, with the title in the T$ array that corresponds to the Row of the chosen song.

Line 440 starts the analyze routine: the Octave default is 0, the duration is 2 (this value will be multiplied by T=200 initialized in line 230; change that value to change the tempo) and the note counter K is set to zero. Tex then reads all characters in the song string into C$; knowing its ASCII value, it refers to the PS array to find the X pointer explained above, and the programs flows to various lines according to that pointer. If it is between 1 and 13, we deal with a note (A, B, C etc), dealt with in line 460: the total note counter K is incremented, and S(K) takes a value which is the note pointer less one multiplied by the octave value O1 (which is zero for the default lower octave, 12 for the upper one, see line 510); if the pointer X=1, we have a rest, and S(K) takes the value of zero for both octaves.

The other characters are dealt with by lines 470-510: a sharp or a flat will augment or diminish the note pointer by one (line 480), a period indicating a dotted note will multiply the duration by 1.5 (490), a digit character will modify the current duration (500) and if Tex encounters a "+" or a "-" it will change the octave accordingly (510). At then end of this text I will explain how to read each character is the song strings.

All that time, our disk was moving towards the turntable; since we don't want to miss it, line 520 checks its position until it reaches pixel column 218, then it stops it. But what if the song is so short that the disk is still too far? After all the song has been processed, line 540 will continue the move until the disk reaches column 218; when it does, it changes its MOTION to a positive vertical value (to move down) and a zero horizontal value. Line 550 checks its descent until it reaches the turntable (pixel row 170); it then DELete the moving disk SPRITE and places an horizontal one on the turntable.

We can finally hear the song! Line 580 replaces the song menu with an empty character, and line 590 displays the chosen title in the jukebox, with a reminder that you can stop play with any key. Line 620 makes the musical notes visible with random colours. Then each note of the song is played, with a CALL KEY between each to allow you to press the "anykey". If you press it, line 650 makes the musical notes invisible again and sends you back to line 280 to display the song menu.



In the line that plays the song (630), the duration is the tempo T multiplied by the value analyzed for each note, and the frequency is that which is pointed to by the note array S(X); while each note is played, the empty character that erased the menu is coloured at random, so you get the effect of an old Wurlitzer jukebox! When you type the DATA, be extra careful with the song strings; each song is one complete string, DO NOT include the spaces on the second line of some, it is there only to make the program listing easier to read. If you make a mistake, the program will still work OK, but the song will not play well to your ear!

Now the details of the song strings, which you can use to play the songs on a piano, or to add more songs, or to extend the ones already in (I extended AULD LAND SYNE in two minutes); just be careful that your song has 60 notes or less, or modify the DIM S() and D() in line 120 accordingly. Each time you change the octave or the duration of the notes, it will be the default for the next ones, until you change them again (see AULD LAND SYNE for an example of both).

OCTAVES (enter before the notes): + for higher, - for lower; Tex assumes the lower octave unless changed in the string.

NOTES: letters A-G, R for a rest (for a C at the top of the current octave, use B#; if you are in the lower octave, you can also use +C).

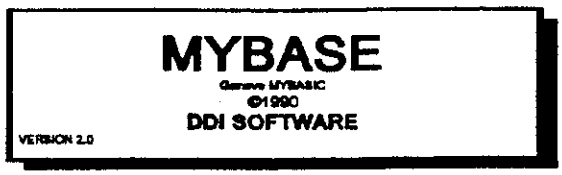
ATTRIBUTES (enter after the note): # for sharp, % for flat.

DURATION (enter after the note): 1 for whole note, 2 for half note, 4 for quarter note, 8 for eighth note; the first note of any song is assumed to be a quarter note unless you change it. To get a dotted note, put a period after the note and any other attribute (for example, a dotted, C-sharp quarter note is entered as "C#4.").

NEW RELEASE NEW RELEASE NEW RELEASE NEW RELEASE NEW RELEASE

A MYBASIC PROGRAM FEATURING A PROGRAM DIRECTOR, WORD PROCESSOR, SPREADSHEET, GRAPHIC LABELER, DISASSEMBLER AND A 6 FUNCTION CALCULATOR. ALL FEATURES ARE IN MEMORY AND CAN BE SELECTED FROM MENU WITH A SINGLE KEYSTROKE.

PRICE $25.00 Postpaid in U.S.

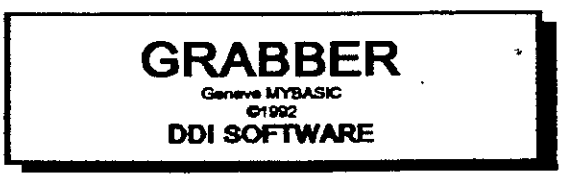


A DATABASE PROGRAM FEATURING INDEXING ANY OF 10 FIELDS, SEARCH FOR ANY SPECIFIC DATA OR RANGE OF DATA, FILTER ALLOWS YOU TO SELECT WHAT DATA YOU WANT DISPLAYED OR PRINTED, EXCLUDING ALL OTHER DATA. COMES SETUP AS A MAILLIST BASE AND CAN BE QUICKLY CHANGED TO ANY KIND OF DATABASE.
PRICE $25.00 Postpaid in U.S.

A CROSSWORD PUZZLE CREATOR OR SOLVER COMES WITH PUZZLES TO SOLVE.
PRICE $15.00 Postpaid in U.S.

ALSO AVAILABLE ARE SOLVUM1, SOLVUM2, TWO DISKS OF PUZZLES TO SOLVE.
PRICE $5.00ea when purchased with MYPUZZELL else $10.00ea.

A PROGRAM PACKAGE COMPRISED OF AN EDITOR VIEWER AND CONVERTER OF ICONS PORTED OVER FROM PC. ALSO HAS OPTION TO ALLOW YOU TO CREATE YOUR OWN DESIGNS. PACKAGE INCLUDES OVER 200 ICONS.
PRICE $25.00 Postpaid in U.S.



A PROGRAM THAT GRABS A MYBASIC SCREEN AND SAVES IT IN ASSEMBLY LANGUAGE OBJECT CODE OR ASSEMBLY LANGUAGE SOURCE CODE FORMAT. ALSO CAPTURES REDEFINED CHARACTERS IF USED. DESIGN MDOS SCREENS IN MYBASIC. EASY TO USE.
PRICE $30.00 Postpaid in U.S.

A 9 OR 18 HOLE GOLF GAME WITH GRAPHIC SCREENS. WATCH OUT FOR THE TREES, WATER AND SANDTRAPS. BOTH PRO AND AMATEUR LEVELS. PAR IS TOUGH 72.

PRICE $20.00 Postpaid in U.S.

TIPSPAINT
Genius MYBASIC
©1990
DDI SOFTWARE

LOAD AND PAINT A TIPS GRAPHIC THEN SAVE AS A PICTURE. ALSO CAN PRINT YOUR PICTURE IN 8 DIFFERENT SIZES. COMES WITH SAMPLE PICTURES AND INFO ON HOW TO INCLUDE PICTURES IN YOUR OWN PROGRAMS.

PRICE $25.00 Postpaid in U.S.

PXLGRABBER
Genius MYBASIC
©1992
DDI SOFTWARE

A PROGRAM THAT GRABS A MYBASIC SCREEN PIXEL BY PIXEL AND SAVES IT IN ASSEMBLY LANGUAGE OBJECT CODE OR ASSEMBLY LANGUAGE SOURCE CODE FORMAT.
PRICE $25.00 Postpaid in U. S.

MAIL ORDERS TO: JIM UZZELL 615 ASHE ST. KEY WEST. FL. 33040

DDI SOFTWARE
615 ASHE ST.
KEY WEST, FL. 33040-7110

NOTE:
THE FOLLOWING IS NOW AVAILABLE AS OF 10-14-92 FROM 9640 NEWS P. O. BOX 752465 MEMPHIS. TN. 38175

MDOS H VERSION 1.21	$2.50
MDOS F VERSION 1.21	$2.50
MYBASIC 3.00	$2.50

92-93 FAIR SCHEDULE

DEC 19, 1992 (SAT) MARLBOROUGH, MA. ROYAL PLAZA TRADE CENTER/HOTEL. 600 TABLES. MASS.PIKE TO I-495 TO EXIT #24B -1/2 MILE ON RIGHT SIDE. KGP.

1993

JAN 3, 1993 (SUN) BAYSIDE, NY. 10AM-3PM $6.00. CALL TRI-STATE COMPUTER FAIRS FOR MORE INFO. TSCF.

>>>> FEB 13/14, 1993 (SAT/SUN) FEST WEST 93. SALT LAKE CITY, UTAH. MORE INFO CALL SALT FLATS BBS (801) 394-0064. HoJo Hotel (801) 521-0138 or (800) 366-3684.

>>>> MAY 14/15, 1993 (FRI/SAT) LIMA MULTI USER GROUP CONFERENCE. OHIO STATE UNIVERSITY LIMA CAMPUS. INFO:CALL CHARLES GOOD. (419) 667-3131.

LITI 99ERS NEWSLETTER IS NOT RESPONCIBLE FOR CANCEL-ATIONS. CALL THE NUMBERS BELOW TO VERIFY TIME AND DATES BEFORE YOU GO.

* Ken Gordon Productions (KGP) SHOWS COST $8.00-$1.00 discount cards are sent to those people on there mailing lists. Call (800)631-0062 OR (908)297-2526 for info. (rev 9/13/92).

* Tri-State Computer Fairs (TSCF) SHOWS COST $6.00-$1.00 discount cards available by mail. Call Robert Barlow (201) 533-1991 for info. (rev 11/7/92).

Don't miss the bargains on computers, software, IC's, peripherals, printers, monitors, parts, supplies and books.

XB MISCELLANY #12
By Earl Ragusa

FILES, FILES, FILES and STILL MORE FILES

When I was a young boy, my father was a blacksmith and a farrier (that's a $25 word for a blacksmith horseshoer). Then a FILE was either a Mill File (smooth) or a Double Cut Bastard (fairly course). There was also the "file" that my father used to finish trim the horse's hoofs, called a "Horse Rasp". It had a very coarse side, and the other side was a little rougher than the D C Bastard. I used it for shaping wood. In school, we had fire drills, where I learned what "Single File" meant. Later, I found out that the administrators kept "files" on the students. When I grew old enough to get a job, found out there were "Payroll Files"

That was my education on files until I went to UCI, circa 1970, and learned some computer programming in Fortran and BASIC. Then I discovered that just about everything having to do with the computer required a "file". If you wrote a program, that was a "Program File", and many computer programs wrote and read "Data Files". Now I find that our 99/4A is pretty much like those computers, only better.

Program and data files are what I shall talk about this time. What I say here is a compilation of what I have read in many newsletters, and from personal experience with the TI 99/4A, not the least of which were the XB and E/A Manuals. hereinafter, EXTENDED BASIC and ASSEMBLY LANGUAGE will be just XB and A/L. BASIC will mean TI-BASIC unless otherwise noted. A partial list of some newsletters which have published good file info are: the UGOC ROM, LA 99ers Topics, Long Island 99er, St Louis Computer Bridge, Greater Akron 99er, Columbus's Spirit of 99, Cleveland Area UG newsletter, Birmingham BUG etc, etc, etc.

I suppose "program" files, as the disk cataloggers call them, are the most common, for the 99/4A they can be in many forms and languages. In BASIC and XB they are largely distinguished by the size. 45 sectors is the largest BASIC file that can be loaded. XB size can reach about 50 sectors, after that they become INT/VAR 254 files, requiring expansion memory. Most BASIC programs will run in XB unless character sets 15 and 16 are used. When you try to run such a file in XB you will probably crash with a "BAD VALUE IN in xxx".

Programs written with XB, using the rules of BASIC will run in BASIC, but a normal XB program will often crash with a "FOR NEXT NESTING in xxx" etc. If you list the program you may see a lot of gibberish. because BASIC cannot interpret a double colon correctly, or the DISPLAY and ACCEPT, commands among other things.

Program files may be identified by their size as follows:

<33 Sectors: Try in order, BASIC, XB, A/L

33 Sectors:
Probably an assembly language program, especially if there is another file with the same name, but the last letter is the next letter of the alphabet.
Try to RUN it using the E/A cartridge LOAD and RUN Option.

34 Sectors:
These are probably GRAM-U-LATOR or GRAM KRACKER files. They will end in numbers from 1-7. You need a GRAM device to run them.

>34 Sectors: First try it in BASIC or XB, it may be necessary to free up memory with CALL FILES MEM OLD DSK1.Name RUN. This could also be a FORTHSAVE file, and can only be run with the Forth kernel, see DIS/FIX 80 below.

52 Sectors:
Tunnels of Doom files usually use this format.

54 Sectors:
Scott Adams Adventure series uses this format.

Other Program Files that won't RUN:
Likely a data file for another program, don't erase it, you might find that some other program won't run without them.

File parameters other than length are Internal (INT), Display (DIS), Fixed (FIX), and Variable (VAR). The latter has a record length associated with it.

Program Files of major interest are as follows.

INT/VAR 254 ...

These are XB programs, you must have the memory expansion installed. They are executed with RUN "DSKx.Name". Good programmers place a file named LOAD on the disk, and when XB is selected, it automatically runs and hopefully executes a DIRectory or MENU program for you to select what is to be run.

DIS/VAR 163 ...

Most likely an XB MERGE format program. This is used for both programs and subprograms. They may be loaded with MERGE "DSKx.FileName". They may be loaded into empty memory, (ie after NEW) or with a program already in memory. If any of the line numbers in the merge file are the same as those already in memory, they will overwrite memory lines. A merged subprogram will not RUN, it must be CALLed from within a running XB program.

DIS/FIX 80 ...

These are assembly language programs which can be RUN with the E/A cart, MiniMem, FW etc. One of these is Forth, which will automatically start running when loaded with DSK1.FORTH. When asked for the filename, enter DSKx.Name and press ENTER. Sometimes they will load and start running, like Forth, but more likely you will be asked for a file name again, enter the additonal files, if any, else just press ENTER. Next you will be asked for Program Name. If you have no other info, try START, BEGIN, FIRST, RUN, GAME, or even LOAD. Else use the FW loader, it will suggest names it reads in the file. If all else fails, one can use a sector editor to read some of the names in the last couple of sectors of the file. Personally, I believe that if the programmer makes you go to all this trouble, "to heck with it."

There are three forms of asembly language programs, TAGGED OBJECT, COMPRESSED TAGGED OBJECT, AND MEMORY IMAGE.

TAGGED OBJECT files are stored in DIS/FIX 80. They are in HEX. They are loaded and run as above. Can be loaded via XB or (TI-BASIC using the E/A or MM modules) using CALL LOAD statements.

COMPRESSED TAGGED OBJECT files are like the above, except that they it can not be loaded from XB. Both forms are are produced from the same E/A source code.

MEMORY IMAGE files are the most compact of the assembler programs, and can be stored and loaded from cassettes. They are loaded from disk with E/A option 5 or TIW option 3. They are fast loading and auto-starting. There is a size restriction of 2400 bytes, but larger programs can be loaded as multiple files. The loader looks for files whose last character is one greater than the previous. For example GAME, GAMF, GAMG.

from LITI

XB MISCELANY #13
By Earl Raguse

OH GREAT! STILL MORE FILES YET!

This column will finish up on the file situation. Last time, program files were covered to the extent of my knowledge. This article covers DATA FILES that are used by various programs. Most of this was copied from an article in the Greater Akron 99er by Jerry Keisler.

DATA FILES

Files such as INT/FIX 108, INT/VAR 128, INT/VAR 64 and some program files are data files used by the program on the disk. Data files will not run, and should be left on the disk with the other programs. If you just wish to get rid of deadwood, copy the disk, then delete the file and see if all programs will run. If so "you did good".

QUICK REFERENCE for PROGRAM FILES

Type	Size	Info or Try
BS	---	XB, BASIC, E/A, CART
PG	33	E/A #5, BOOT, FW
PG	34	GROM SIMULATOR
PG	52	TUNNELS OF DOOM
PG	54	ADVENTURE
DV80	---	TIW, FW, ETC
DV163	---	XB MERGE
DF80	---	E/A 3# (ASSY, FORTH)
DF128	---	ARCHIVER, FORTH
IV254	>45	XB, TIPS Fonts (no run)

SOME OTHER FILES

TI-BASE

Ext	Type	Description
/P	IF255	Program
/H	DV80	Help
/C	DV80	Command
/C	DV40	Command
/D	IF	Data
/S	IF255	Data Base Structure

TI-ARTIST

Ext	Type	Description
_C	25PG	Pictures Color
_P	25PG	Pictures Pattern
_F	DV80	Character Font
_S	18DV80	Slides
_I	DV80	Instance
_V	DF12	Vector
_M	DV254	Movie

TIPS

Ext	Type	Description
	IF53	Picture (TXT)
	IF16	Name text (XXX)
	DV250	Spooled Graphic
	IV254	TIPS Fonts

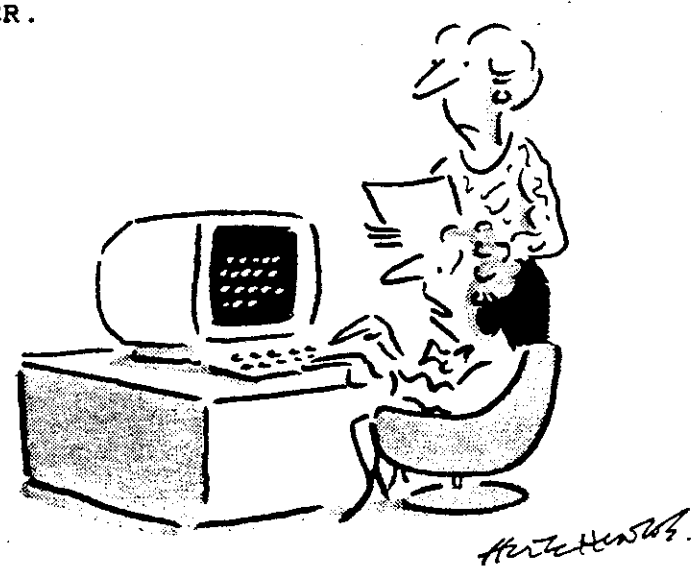
That's about it.

XB MISCELLANY #12 continued from page

DIS/FIX 128 ...

These are probably Forth screens which must be loaded and compiled with Forth. Do not confuse with Archived files. These files are identified as SYS-SCRNS. One executes a loaded and compiled Forth program by entering the key word. Most probably the last word defined on the last screen loaded. Good Forth programs will prompt you, and you should not even need to know that it is written in Forth. Some Forth disks will load from XB like any other auto-loading XB disk with a LOAD program.

That concludes the list of program files in the normal sense. I will continue with the various forms of data files.
ER.



"I thought this new system was supposed to give us more accuracy, flexibility and better information flow. It keeps telling me, 'Try again, better luck next time!'"

from LITI

XB MISCELANY #14
By Earl Raguse

THE MYSTERY OF ON ERROR AND RUN

When playing around with my DIRectory program, I discovered what appeared to be a strange operation of ON ERROR. I frequently put a directory on a disk when it is only partially filled. I also sometimes give the disk to people before the disk is complete. The tendency is for people to select a blank line in the directory. That is, I am sure, just curiosity as to what would happen, as opposed to being down-right mean, to see if they can make my program crash.

When a line is selected from the displayed directory, it in turn causes the program to go to a specific line to RUN "DSK1.xxxxxx". If the directory line is blank, so is "xxxxx". Now XB does not like to be told to RUN a blank program. An erroneous one is just an error, but a non-existent or blank program just puts XB into a tizzy. I used to put RUN "DSK1.DIR" on all the blank lines, so the program would re-run itself. But that was not a good solution, because that required users of a blank DIRectory to overwrite parts of the program. Now that is very scary to some people. They do not mind filling in blanks, but overwriting, UGH!

Then I got a brilliant idea, why not use ON ERROR nnn. Then when someone selected a blank line, and the program tried to RUN the blank program, the error would cause XB to go to line nnn, where a sarcastic message would be displayed about people who select blank lines from a directory.

I thought I would even make this a subprogram. IT DOES NOT WORK! WHY? Because when XB executes RUN, it apparently wipes all memory of subprogram locations and a whole host of other things. You can't even do a CALL CLEAR after that kind of a crash. Normally when RUN is successfully executed a program is loaded, all the required utilities are put in place and the program is run as expected. But, if no program is available to run, XB appears to get lost. Page 162 of the XB manual carries a momentous 59 word discussion of RUN. At least 25 percent of those 59 words are three letters or less. There is absolutely no mention of what actually goes on. This is one of the few times, however, that a good example is given, and all is well if there are no errors, otherwise CRASH!!!

My final solution to the problem was to abbreviate my blank RUN lines from RUN "DSK1." to RUN. How does that work? Well in a program you may execute RUN as a program command, and it simply causes the program in memory to be run again. No harm, No foul. Sometimes this is an effective way to clear a large matrix of number or string data, for another run.

I do not wish to take the space to publish the actual program, but I have included here a short program called ERRORTEST to demonstrate what I have been talking about. The program offers two paths A and B. Enter the program as listed, with lines 160 and 180 having the comment mark to inactivate them, the program will work sort of OK. But if you remove the comment mark from line 160 or 180 or both, to activate them, and select A, you will get an error message. If you select B the program still cycles just fine.

If you select A, the program tries to execute line 150, this produces an error, this directs the program to line 160. If 160 is active, you will crash and be told that the subprogram, ie CALL CLEAR, could not be found. Line 180 produces a syntax error message because it cannot execute the FOR-NEXT loop of line 210, even though it has no syntax error. Try them one at a time. I have determined also, that IF THEN statements will not work either.

Note what happens if you select B, the program executes line 200, displaying a friendly message, while it executes the FOR-NEXT loop of line 210, then executes RUN, to repeat the program. This does not depend on whether lines 160 and 180 are active or not.

```
100 ! SAVE DSK1.ERRORTEST
110 ! By Earl Raguse 4/92
120 ON ERROR 160
130 DISPLAY AT(10,1)ERASE AL
L:"     PRESS A FOR PATH #1,
        PRESS B FOR PATH #2"
140 CALL KEY(3,K,S):: IF S<1
 THEN 140 ELSE IF K=ASC("A")
 THEN 150 ELSE IF K=ASC("B")
 THEN 200
150 RUN "DSK1.       "
160 !CALL CLEAR
170 DISPLAY AT(12,1)ERASE AL
L:"ERROR, ERROR, YOU BIG  DU
MMY"
180 !GOSUB 200
190 GOTO 120
200 CALL CLEAR :: DISPLAY AT
(20,1):"   NO PROBLEM, Pause
 here              then R
UN" :: GOSUB 210 :: RUN
210 FOR T=1 TO 300 :: NEXT T
::RETURN
```

From the TI-echo...

Here is an assembly program for the TI-99/4A written by Jonathan D. Guidry, that turns the cassette port on and off and the audio port on the cassette port on and off.

```
        DEF MTRON,MTROFF
        DEF AUDON,AUDOFF
STATUS  EQU >837C
GPLWS   EQU >83E0
MTRON   CLR R12
        SBO 22
        JMP QUIT
MTROFF  CLR R12
        SBZ 22
        JMP QUITAUDON
         CLR R12
        SBZ 24
        JMP QUIT
AUDOFF  CLR R12
        SBO 24
        JMP QUIT
QUIT    CLR  R0
        MOVB R0,@STATUS
        LWPI GPLWS
        B    @>0070
        END
```

-WP♦

From: Tim Tesch
To: Anyone

Subject: Ansi Programs

If anyone is interested, I have modified two versions of Mass Transfer to display ANSI-graphics. Additions were made to version 4.3 for the TI, and versions 3.9 (80 columns) and 4.3 (40 columns) The TI version has additions which will allow you to toggle the ANSI graphics on or off from the main menu. The other two versions for the Geneve/80 column card were not as easy to modify, so I have created a small object code (D/F80) file which does the extra work. One nice thing about these programs is that they will capture the graphics in the buffer/log and will print them, or even SAVE them to disk for later use! Not even Telco can do this, it just saves asterisks '*'s in place of the graphics. Once saved, these files can be used and edited by using my ANSI-TOOLS program. The programs do not as of yet recognize the ansi command structure but some work is being done to at least include some of the basic ESCape codes for positioning the cursor, etc.

If you're interested in the Mass Transfer versions, send a blank disk or $3.00 to cover diskette costs and mailer to:

Tim Tesch
4346 North 88th ST
Milwaukee, WI 53222

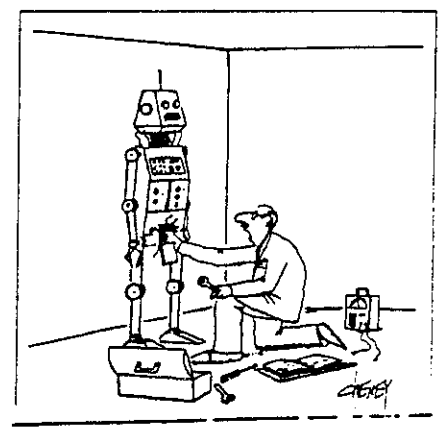
I will also sell copies of ANSI-TOOLS to anyone wanting one for $13.00 (covers disk & shipping) which will let you create your own ANSI screens or edit those captured with the above versions of Mass Transfer. If you want the modified Mass Transfer files with your registered copy of ANSI-TOOLS, just write a note requesting them.

WHAT IS A MAIL MERGE?

A Mail Merge is a convenient way to mail a form letter to many people without manually typing in the names, addresses, etc. This id extremely useful when various forms of communication must be sent to the same group of people monthly, quarterly, etc. So the obvious thing then would to create the mailing list. Use this... Example:

```
1 Mr
2 Tom
3 Jones
4 2341 Any Street
5 Somewhere, CA 91123
6 Jones family
*
Ms
Jane
Smuthers
7777 Lucky Lane
Sameplays,CA 91119
Smuthers family
*
1 Mrs
2 Eunes
3 Somuch
4 2468 'Preciate Ct.
5 Fairytale, CA 91121
6 Somuch family
*
```

(Press <ENTER> after each line. Yes! the numbers 1, 2, 3, 4, 5, 6 must be entered. Notice the '*' asterisk separating each group of members, TI-Writer recognizes it as a separator.)



Save as DSK2.MYFILE/1

Next, let's create a form-letter. This is the one I created for our August meeting (which Fred Moore put on for me when I realized that I was playing (I'm a musician) on that night.) A typical head- would look similar to this:

```
.FI;AD;LM 8;RM 72          <enter>
.IN +32                       "     (the +32 will print the follow
Chick De Marti                "      ing lines near the center  of
18028 Falda Ave.              "      the page).
Torrance, CA 90504            "
<enter>                       "
August 26 1991                "
.IN +0                     <enter>  This cancels the .IN on line 2
<enter>
<enter>
*1* *2* *3*                <enter>    = Title, F-name,L-name
*4*                           "       = Street address
*5*                           "       = City, State, Zip code
<enter>
Dear *2*,                  <enter>    = F-name
<enter>
```

THE KIDDIE CORNER
by Sue Harper
Pittsburgh Users Grooup

For kids of all ages - a series of articles on how to get started making your own programs.

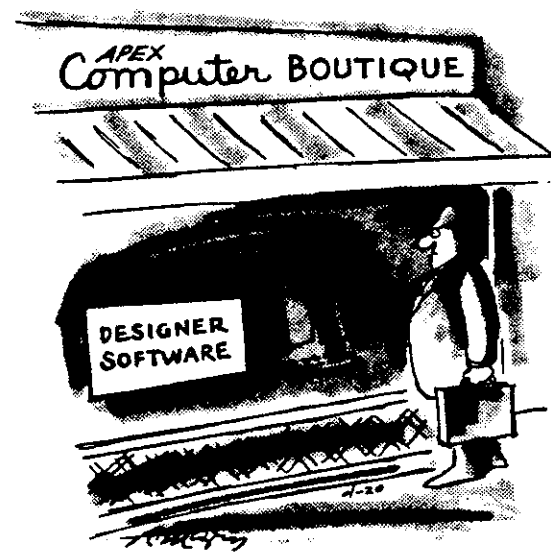
A few months ago I proposed a challenging program for all you enthusiasts to work on. I hope that you have enjoyed working on the program, or on one that you thought up yourself.

For those of you who want to see the completed program, I must admit that I have busied myself with other things and do not have the program worked out. However, here are the basic necessities:

The beginning would have a title, a list of choices, and a way to let the user choose:

```
100 CALL CLEAR
110 REM THIS PROGRAM IS WRITTEN IN BASIC
120 REM SPEECH MODULE AND TEII REQUIRED
130 PRINT TAB(20);"TI WORLD"::::::::::::::
140 FOR WAIT=1 TO 250
150 NEXT WAIT
160 PRINT"PRESS ANY KEY TO BEGIN"
170 CALL KEY(0,K,S)
180 IF S=0 THEN 170
190 REM THIS SECTION GIVES THE CHOICES
200 CALL CLEAR
210 PRINT"PRESS THE LETTER OF YOUR CHOICE"
220 PRINT"A. A SONG"
230 PRINT"B. A QUIZ"
240 PRINT"C. A GRAPHICS DISPLAY"
250 PRINT"D. TO END THE PROGRAM"
260 CALL KEY(0,K,S)
270 IF S=0 THEN 260
280 IF K=65 THEN 330
290 IF K=66 THEN 2000
300 IF K=67 THEN 3000
310 CALL CLEAR
320 STOP
330 CALL CLEAR
340 CALL SOUND(200,392,15)
350 CALL SOUND(100,392,15)
(continue with call sound statements to
play the entire song)
1990 GOTO 200
2000 CALL CLEAR
2010 PRINT" GUESS THE ANSWERS! GOOD LUCK!"
2020 PRINT"I WILL KEEP SCORE FOR YOU!"
2030 LET SCORE=0
2040 LET S=0
2050 PRINT" QUESTION 1: WHAT COMPANY FIRST
DEVELOPED THE COMPUTER CHIP?"
2060 PRINT" A. WESTINGHOUSE"
2070 PRINT" B. TEXAS INSTRUMENTS"
2080 PRINT" C. WESTERN UNION"
2090 PRINT" D. COCA COLA COMPANY"
2100 PRINT TAB(20);SCORE
2110 CALL KEY(0,K,S)
2120 IF S=0 THEN 2110
2130 IF K=66 THEN 2170
2140 CALL SOUND(500,-3,15)
2150 PRINT"WRONG ANSWER! TRY ANOTHER
QUESTION!"
2160 GOTO 2190
2170 PRINT"VERY GOOD! TRY ANOTHER QUESTION!"
2180 LET SCORE=SCORE+1
2190 CALL CLEAR
COMPUTER ARE YOU USING RIGHT NOW?"
2200 PRINT" QUSTION 2: WHAT BRAND OF
(continue with questions and answers for as
long as you wish following the above format)
2990 GOTO 200
3000 CALL CLEAR
3010 CALL CHAR(96,"0000FF0FF0FF0FFF")
(continue with call char statements to
define your graphics)
3500 CALL HCHAR(12,12,96)
(continue with call hchar and call vchar
statements to place your graphics on the
screen)
4000 GOTO 200
```

That will - in this condensed form with hints to help you keep building - create the program I had in mind. I hope you can fill in all the blanks. If not, let me know by contacting me, or anyone in the Pittsburgh Users group who can then pass on the message.



COMPARISON CHART FOR 80 COLUMN CARDS
By: Andy Frueh, Lima UG

Feature(s)	Asgard EGI/ Mechatronics	Dijit AVPC	OPA TI-Image Maker
Method of installation	Sidecar	P Box card	Internal board
Modification to console?	No	Some	Some
Highest resolution	512 x 424	512 x 424	512 x 424
Max colors in that mode	16 of 256	16 of 256	16 of 512
Most colors in one mode	256 (256 x 424)	256(256 x 424)	256 (256 x 424)
Ports/outlets	Mouse port #1	Serial port	None
Operating system?	TI's	TI's	OPA OS #2
VDP chip used	9938	9938	9958
VDP memory standard/max	192K	128K/192K	192K
Sprites	Unknown #3	Unknown	32/8 per line
80 column modes	80 col/24 rows #	80/24 #4	80/24, 80/26.5
Composite compatible?	Yes #5	Yes #5	No #6
Total # of modes #7	8	8	8
Approx. price	$250	$220	$179

Thinking about buying an 80 column card, but can't decide which one to choose? This chart may help you decide. No opinions, just facts.

Notes

#1 I'm not sure if Asgard's device (same as the Mechatronics) will also have the mouse port

#2 OPA developed an operating system to cure any software problems. Some programs do NOT operate properly on 9938/58 systems.

#3 I know these devices will allow 8 on one line, but I'm not sure on the maximum number of sprites with the AVPC and EGI. An unmodified TI will give you 28 sprites, 4 on a line.

#4 There may be other modes, but I'm not sure on that. All 80 column programs are designed to use 80x24 mode anyway.

#5 But it is NOT reccomended. An RGB monitor is desirable

#6 There should be available soon a device to let the T.I.M. be composite compatible.

#7 All of these devices have 256 by 192/212/384/424 as well as 512 by 192/212/384/424 modes. The differences are mainly in the total number of colors. Using what is called the YJK System Display, the OPA device can use 19,268 SIMULTANEOUS colors. The other two are limited to 256.

Now some more interesting tidbits. The EGI, although the most expensive in our group, offers a double-edged sword type advantage. All you do is plug it in and go. No set up or modification, no matter how simple. This means any computer can be set up for 80 columns in about 5 seconds. The bad part is that your console will increase in width. Also, some devices seem to have unreliable contacts when you use them "sidecar" fashion. By some devices, I do NOT mean the EGI, just some of the other sidecar devices. Both of these problems can be corrected with a RELIABLE sideport cable.

More interesting facts. Check out this table comparing video formats. It compares two of the best IBM/clone graphics modes with an unmodified TI and a 9938/58 TI.

	IBM CGA	IBM VGA	Norm. TI	9938/58 TI
Best resolution	620x200	720x350	256x192	512x424
Total pixels on screen	128000	252000	49152	217088

That's right. The "super video" TI's are better then a "low res" IBM and almost as good as a top of the line IBM. Need more convincing? Well if you think that the TI has good graphics (very good for a 1979 computer) and feel that these devices offer nothing for you other than a handy $180-250 aid in word processing, consider this. The modified TI's have a graphics screen that has barely over 4.4 TIMES the resolution of a normal console!

DONE

RECIPE OF THE MONTH

BY
DEBBIE GAZSY

BUTTERY CHRISTMAS COOKIES

Ingredients:

6 cups flour
5 eggs
5 tsp Baking Powder
1/2 lb of margarine (softened)
1 1/2 cups Sugar
1/4 to 1/2 cup Orange Juice

Mix the sugar and baking powder with 5 cups of the flour in a bowl. Add eggs and margarine to the mixture and stir till ingredients are well blended. If the mixture is too dry and won't hold together, add some orange juice. Likewise, if the mixture is too wet, add a little flour. When the batter has the right consistency, place some flour on the table and roll the dough out to 1/4 inch thickness. Preheat oven to 375 degrees for ten minutes. Cut cookies out of the dough and place them on a cookie sheet 1 1/2 inches apart. Bake at 375 degrees for 14 to 18 minutes. Cookies are done when the bottoms are brown. Ingredients will make 6 - 8 dozen cookies.

Merry Christmas
from
Gazsy Cookie Jar