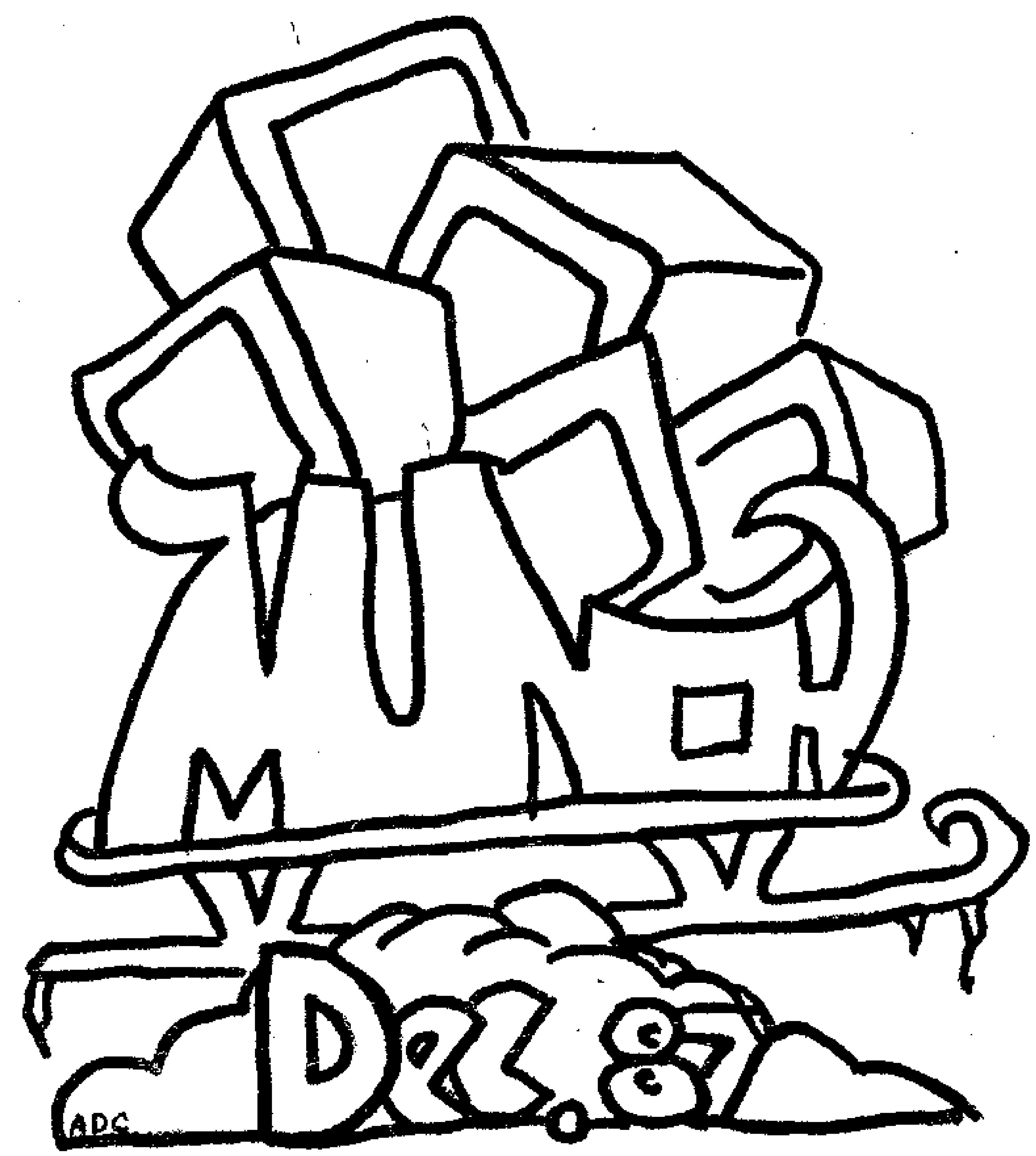
happyholidayshappyholidayshappyholidayshappyholidays

Mass Users of the Ninety-nine and Computer Hobbyists

DECEMBER 1987 Monthly Newsletter   Version 6.12

>>>>>>>>>>>>>>NEW LOCATION!!!!!<<<<<<<<<<<<<<<<

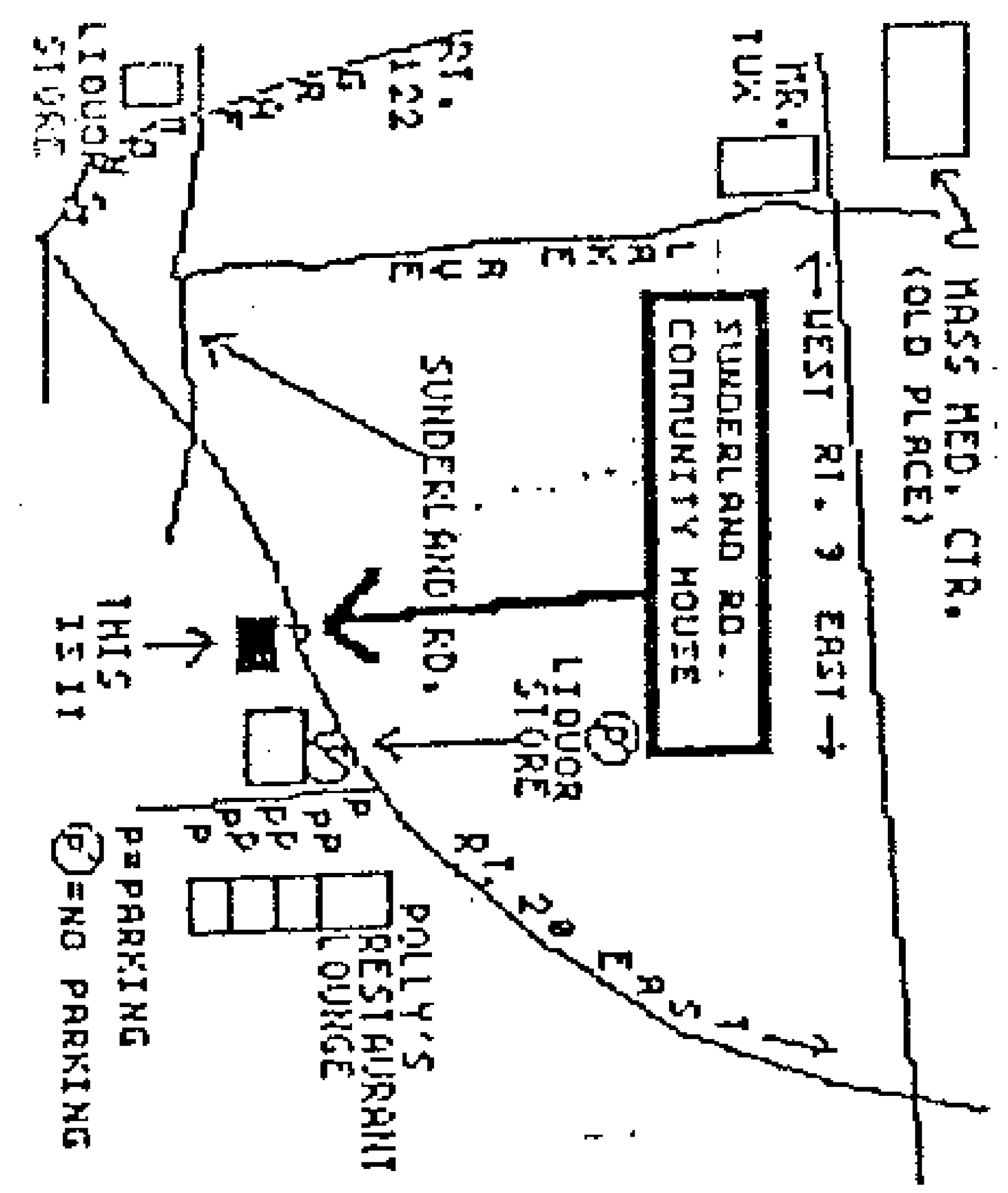


JOIN THE CROWD AT OUR DEC. 8 MEETING!

M.U.N.C.H.
P.O. Box 7193
560 LINCOLN STREET
WORCESTER, MA. 01605

<u>FIRST CLASS</u>
SEASON'S GREETINGS

RENEW!!



2nd
TUES
of the
Month

<u>NEW LOCATION</u>
SUNDERLAND RD
COMMUNITY CTR
ROUTE 20

DECEMBER 8th 87 / IN OUR NEW CLUBHOUSE!

S G
E R
A E
S E
O T
N I
' N
S G
S

MUNCH OFFICERS AND NUMBERS (all in 617 area)

President/Mail	W.C. Wyman	839-4134
Vice President	Hector Beaudreau	
Secretary	Al Cecchini	
Treasurer	Jim Cox	869-2704
Editor/Lugger	Jack Sughrue	476-7630
Adv Prog. Chair	Dan Rogers	248-5502
Forgetter	Ronald Reagan	000-0000
Library	Al Lisa Cecchini	
Software Library	Don Mason	754-6630
	Hector Beaudreau	
Reporter	William Shakespeare	
BBS Hostess	Helen Holmes	
Good Guys	The Members	

LIBRARY NOTICE

PLEASE RETURN ANY ITEMS BORROWED FROM OUR LIBRARY. We are still missing a considerable number of books, tapes, disks, and so on belonging to YOUR CLUB . Do a little clearing around your computer area (or any places you'd be apt to set things aside). If you locate any library materials (or if you'd like to donate any you no longer use) please come with them to the next meeting. We don't care how long you've had them out. There is no fine. But it would be fine if other members could have a chance to borrow these things.

ADVERTISING RATES:

Double Page	*(10.5" by 8")*	*$25.00 per insertion*
Full Page	*(5" by 8")*	*$13.00 per insertion*
Half Page	*(5" by 4")*	*$ 7.00 per insertion*
Quarter Page	*(5" by 2" or*	
	(2.5" by 4")	*$ 5.00 per insertion*

Classified (non-commercial) ads are FREE for MUNCH members.

....RAFFLE....

Our raffles in December will continue to be the exciting and valuable. There are numerous donated gifts of all kinds of software and other items, such as cartridges, tapes, disks of all kinds of things: educational, fun, utility. Get ready for Christmas and Channukah by winning your presents this month at the new clubhouse. Donations happily accepted. Remember: YOU MUST BE PRESENT TO WIN!

DECEMBER SALE [Get some gifts for the whole family!]

Another chance to sell any used consoles, P/Boxes, cards, tape recorders, interface cables, ANYTHING related to your computer system. Also bring any original tapes, cartridges, disks, texts, or other soft/textware. Be prepared to buy a lot and sell a lot. Please come with prices marked on the items. Call Jack Sughrue to let him know what you will be selling. AND get set for the December BIGGIE!

NEWSLETTER

Become Immortal! We are looking for articles, cartoons, love letters, programs, lists, old banana peels: in short, anything from the members which can be printed in our newsletter. Text items preferred on SSSD disk through TIW. Printed items also accepted. Share your interest or expertise with other members. Mail all items by the 3rd Tuesday of each month to Jack Sughrue, Box 459, E.Douglas MA 01516. Disks will be returned at the next meeting.

NEWALS + RENEWALS

NEWALS are $15/year plus a one-time $10 inititation fee (which includes a choice of ANY club disk free); RENEWALS are $15/year. Members have full use of disk/text libraries, free workshops + assistance, 12 full issues of M.U.N.C.H., voting privileges + more! Subscription alone is $10/year. Mail check to address on cover.

ONE(+) LINERS FOR GRAPHING

by Tony Falco

Graphing is an important topic in mathematics education today. Computers and computer graphics will make it an even more important topic in the future. The following one and two (sorry!) liners can hopefully help out some middle school or high school students with some of the more basic concepts.

The first program gives practice with the process of plotting points. At the "X=,Y=" prompt the user enters two numbers separated by commas. The program will show that point if its coordinates will fit on the screen. Bear in mind the all the programs here are low resolution and plot only integer points.

The second one graphs the function Y=10*SIN(PI*X/14). Users can experiment with other functions by simply changing that expression. Try Y=ABS(5-ABS(X)) for example.

The third and fourth programs are very similar. The third one allows the user to experiment with sine waves. At the prompt you enter values for A and B, again separated by commas, and see how these numbers change the period and amplitude of the wave. The last one graphs shapes known as parabolas. At the prompt enter values of A,H, and K, once again separated by commas, and see how these values effect the shape, position and orientation. Values of A between -2 and 2 (decimal fractions are fine) work best. Again bear in mind that with low resolution many points do not get plotted, but this is exactly what one does with paper and pencil. We plot a few points and infer the position of the rest.

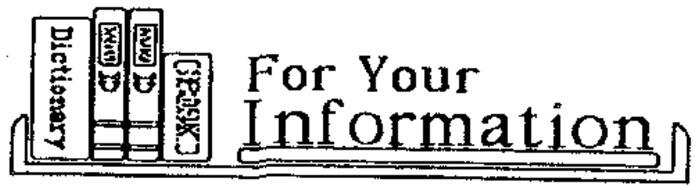
For a neat display run using CALL CLEAR :: RUN.

```
1 FOR D=1 TO 2000 :: NEXT D :: CALL CLEA
R :: INPUT "X=,Y=":X,Y :: IF ABS(X)>15 O
R ABS(Y)>11 THEN 1 ELSE CALL HCHAR(12,1,
43,32):: CALL VCHAR(1,16,43,24):: CALL H
CHAR(12-Y,16+X,30):: GOTO 1
```

```
1 CALL HCHAR(12,1,43,32):: CALL VCHAR(1,
16,43,24):: FOR X=-15 TO 16 :: Y=10*SIN(
PI*X/14):: CALL HCHAR(12+Y*(ABS(Y)<=11),
16+X,42-(ABS(Y)>11)):: NEXT X :: GOTO 1
```

```
1 CALL CLEAR :: INPUT A,B :: CALL CLEAR
:: DISPLAY AT(1,7):"Y="&STR$(A)&"*SIN(2*
PI/"&STR$(B)&")"
2 CALL HCHAR(12,1,43,32):: CALL VCHAR(2,
16,43,23):: FOR X=-15 TO 16 :: Y=A*SIN(P
I*X/B):: CALL HCHAR(12+Y*(ABS(Y)<=11),16
+X,42-(ABS(Y)>11)):: NEXT X :: GOTO 2
```

```
1 CALL CLEAR :: INPUT A,H,K :: CALL CLEA
R :: DISPLAY AT(1,7):"Y="&STR$(A)&"(X-";
STR$(H);")^2+";STR$(K);
2 CALL HCHAR(12,1,43,32):: CALL VCHAR(2,
16,43,23):: FOR X=-15 TO 16 :: Y=A*(X-H)
^2+K :: CALL HCHAR(12+Y*(ABS(Y)<=11),16+
X,42-(ABS(Y)>11)):: NEXT X :: GOTO 2
```



NAMES AND NUMBERS

Software Center
2105 Post Rd
Warwick RI
738-9800
Computer supplies-some TI

Software Connections
101 West Natick Rd
Warwick, RI
738-3430
Across from Warwick Mall
Computer Supplies-Limited TI

Ken Gilchrist
PO Box 531
Dedham, Ma 02026
(617)329-5545
Like going to a TI Flea Market

Paper Works
544 West Ave
Pawtucket, RI
728-0414
Great Prices on paper,labels,etc

Jabbour Electronics
345 Fountain St
Pawtucket, RI
728-4600
Computer chips, electronic parts, etc

MICROPENDIUM
PO Box 1343 Round Rock, Tx 78680
(512)255-1512
Magazine devoted to TI

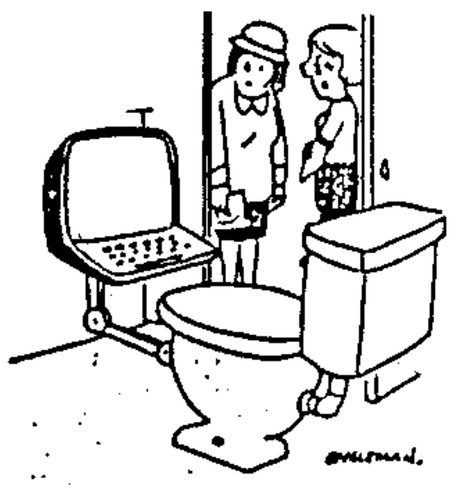
Tenex Computer Express
PO Box 6578
South Bend, In 46660
1800-348-2778
TI Mail order supplies

Texaments
53 Center St
Patchogue, NY 11772
516-475-3480
TI ARTIST and

CSGD

Tex Comp
PO Box 33064
Granada Hills, Ca 91344
1-818-366-6631

Myarc
201-766-1700



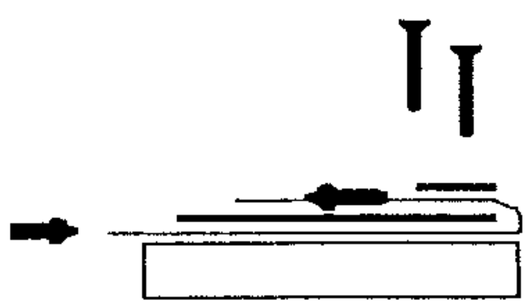
Most people read in the bathroom. Ray is different

Bruce's Computer Magic

by Bruce Kalver- North Eastern 99ers

LABEL PEELER

HERE IS A HANDY DEVICE TO PEEL OFF LABELS. IT WORKS SIMILAR TO THE WAY AVERY LABELS COME OUT OF SMALL BOXES. THE BASE COULD BE CLAMPED ON A TABLE USING C-CLAMPS.



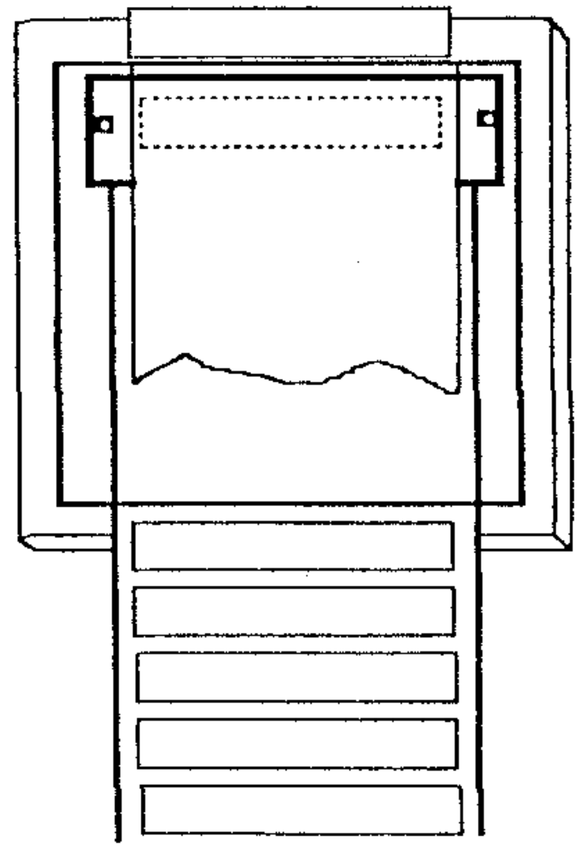
LABELS GO THROUGH THE BOTTOM PLATE ... AROUND ...THEN BACK ABOVE THE PLATE

THE LABELS WILL POP OFF AND STICK TO THE VERY TOP PLATE

THE PLATES ARE MADE FROM VERY THIN METAL OR FORMICA

THE BASE IS A BLOCK OF WOOD

THE SCREWS CAN BE LOOSENED OR TIGHTENED TO GET THE RIGHT TENSION.



THE LABELS ARE FIRST THREADED IN FACE DOWN

TI TRIVIATA

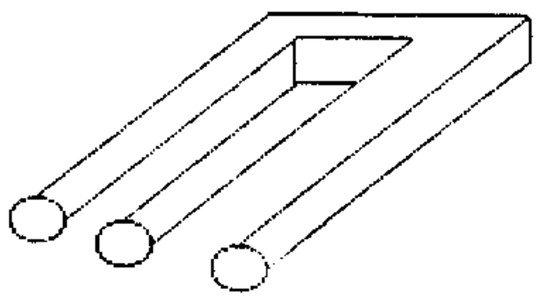
- Maximum # of characters in TI-WRITER: 23,000
- Highest Line Number in a program: 32767
- Frequency Number for Middle C: 262
- Where BASIC was invented: Dartmouth College
- What ASCII stands for: American Standard Code for Information Interchange
- Less than 8% of freeware users ever pay the donation
- The TI is in more homes than any other computer. We are not saying "most used", we are just saying in the homes. Most are collecting dust.
- Micro second: one millionth of a second
- Millisecond: one-thousandth of a second

PROGRAM OF THE MONTH

by Bob August (BUG News)

One year on Television there was an ad for a computer which had a Santa on the screen. You may have seen it. It was all programmed with blocks of color squares and I figured I that would be easy to program, so I wrote the program below in BASIC. I used the following letters to represent the different colors.

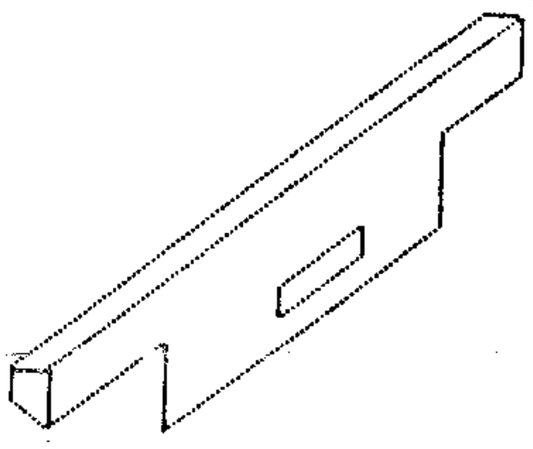
B = Char 128 = Black
BB = Char 129 = Belt buckle
BU = Char 43 = Buttons
E = Char 96 = Eyes
M = Char 42 = Mouth
O = Char 120 = Orange
R = Char 104 = Red
W = Char 112 = White



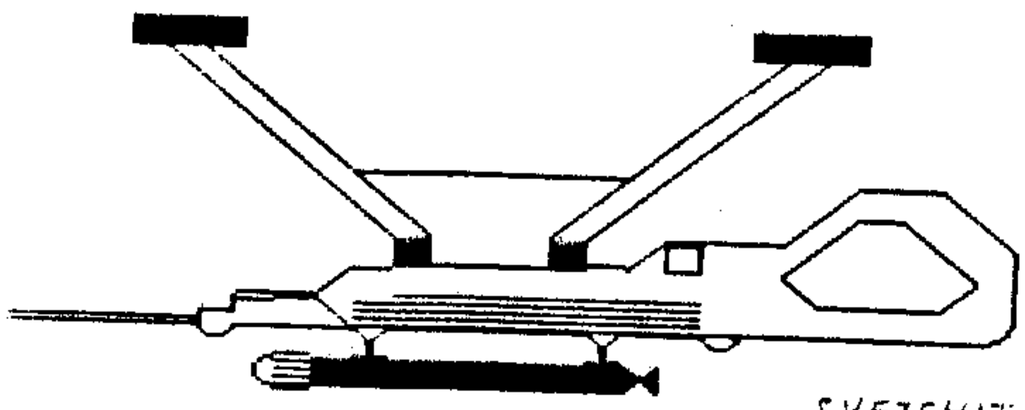
SKETCH IT

Here is the program:

```
100 REM <<SANTA>>
110 REM BY R.W. AUGUST
120 CALL CLEAR
130 CALL SCREEN(5)
140 M=42
150 BU=43
160 E=96
170 R=104
180 W=112
190 O=120
200 B=128
210 BB=129
220 A$="FFFFFFFFFFFFFFFF"
230 CALL CHAR(R,A$)
240 CALL CHAR(W,A$)
250 CALL CHAR(O,A$)
260 CALL CHAR(B,A$)
270 CALL CHAR(M,"FFFFC3C3C3C
3FFFF")
280 CALL CHAR(E,"FFFFFFE7E7F
FFFFF")
290 CALL CHAR(BU,"0000183C3C
18")
300 CALL CHAR(BB,"FF81818F81
8181FF")
310 CALL COLOR(2,2,7)
320 CALL COLOR(9,6,2)
330 CALL COLOR(10,7,1)
340 CALL COLOR(11,16,1)
350 CALL COLOR(12,10,2)
360 CALL COLOR(13,2,16)
370 CALL HCHAR(2,16,W)
380 CALL HCHAR(3,15,R,3)
390 CALL HCHAR(4,15,R,3)
400 CALL HCHAR(5,14,R,5)
410 CALL HCHAR(6,14,W,5)
420 CALL HCHAR(7,14,O,5)
430 CALL HCHAR(7,15,E)
440 CALL HCHAR(7,17,E)
450 CALL HCHAR(8,13,O,7)
460 CALL HCHAR(8,16,R)
470 CALL HCHAR(9,9,W,2)
480 CALL HCHAR(9,13,O,7)
490 CALL HCHAR(10,9,W,2)
500 CALL HCHAR(10,13,W,7)
510 CALL HCHAR(10,16,M)
520 CALL HCHAR(11,9,R,2)
530 CALL HCHAR(11,14,W,5)
540 CALL HCHAR(12,9,R,2)
550 CALL HCHAR(12,13,R,7)
560 CALL HCHAR(12,15,W,3)
570 CALL HCHAR(13,9,R,12)
580 CALL HCHAR(13,16,W)
590 CALL HCHAR(14,10,R,12)
600 CALL HCHAR(14,16,BU)
610 CALL HCHAR(15,11,R,11)
620 CALL HCHAR(16,11,R,11)
630 CALL HCHAR(16,16,BU)
640 CALL HCHAR(17,11,R,11)
650 CALL HCHAR(18,12,W,9)
660 CALL HCHAR(18,16,BB)
670 CALL HCHAR(19,12,R,9)
680 CALL HCHAR(20,13,R,3)
690 CALL HCHAR(20,17,R,3)
700 CALL HCHAR(21,13,R,3)
710 CALL HCHAR(21,17,R,3)
720 CALL HCHAR(22,13,W,3)
730 CALL HCHAR(22,17,W,3)
740 CALL HCHAR(23,11,B,5)
750 CALL HCHAR(23,17,B,5)
760 CALL KEY(0,K,S)
770 IF S=0 THEN 760
780 END
```



SKETCH IT



SKETCH IT

DISK TO PRINTER PRINT

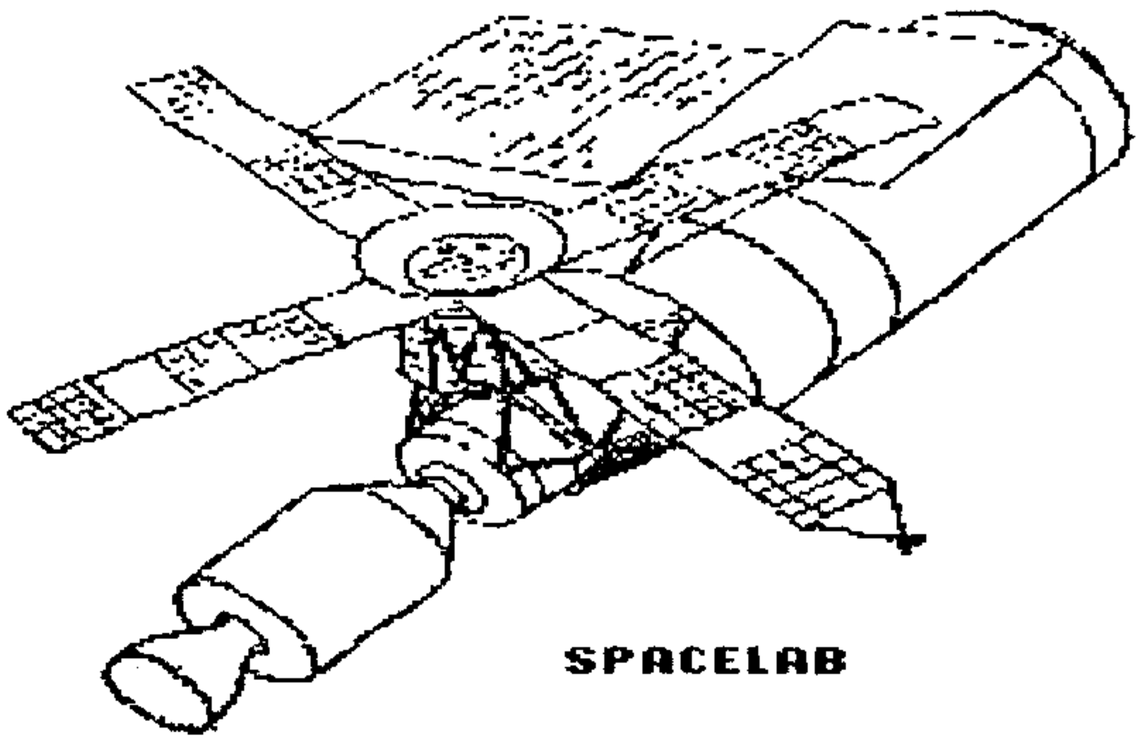
The January 1987 MICROpendium contained a program attributed to Bob Sims of the Nor-Cal TI users group. This program is called PRINTCOPY. It prints D/V80 files directly from disk to printer in an equivalent manner to TI Writer's editor. Changing the LINPUT in line 130 to INPUT will allow its use with internal type files.

The program requires Extended BASIC.

```
80 ! PRINTCOPY
90 ! BY BOB SIMS
100 INPUT "PRG NAME":PR$
110 OPEN #3: "P1O"
120 OPEN #1: PR$
130 LINPUT #1: A$
140 PRINT #3: A$
150 IF EOF(1) THEN 170
160 GOTO 130
170 CLOSE #1 :: CLOSE #3
```

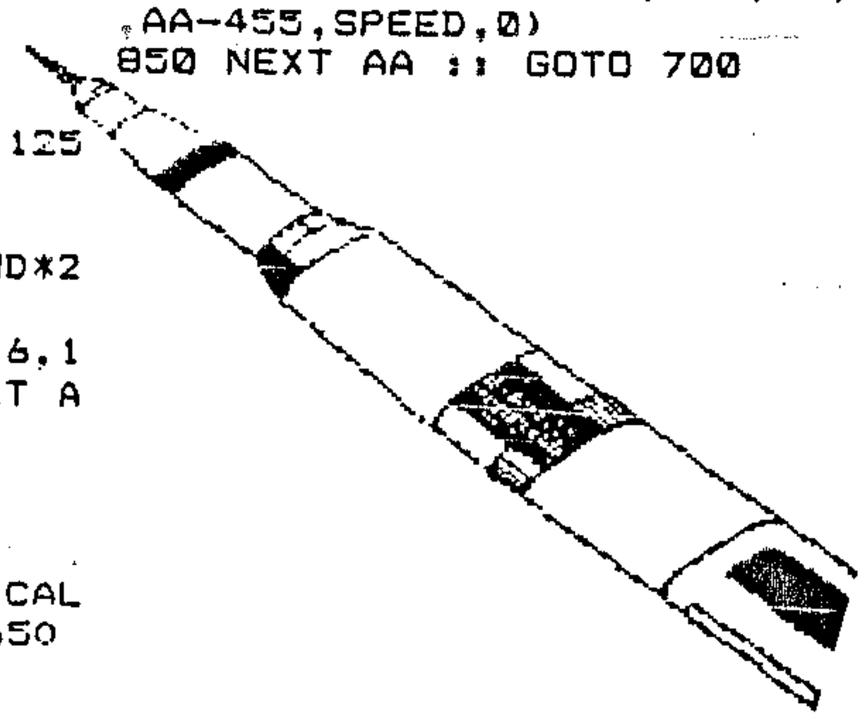
(from LA TopIcs)

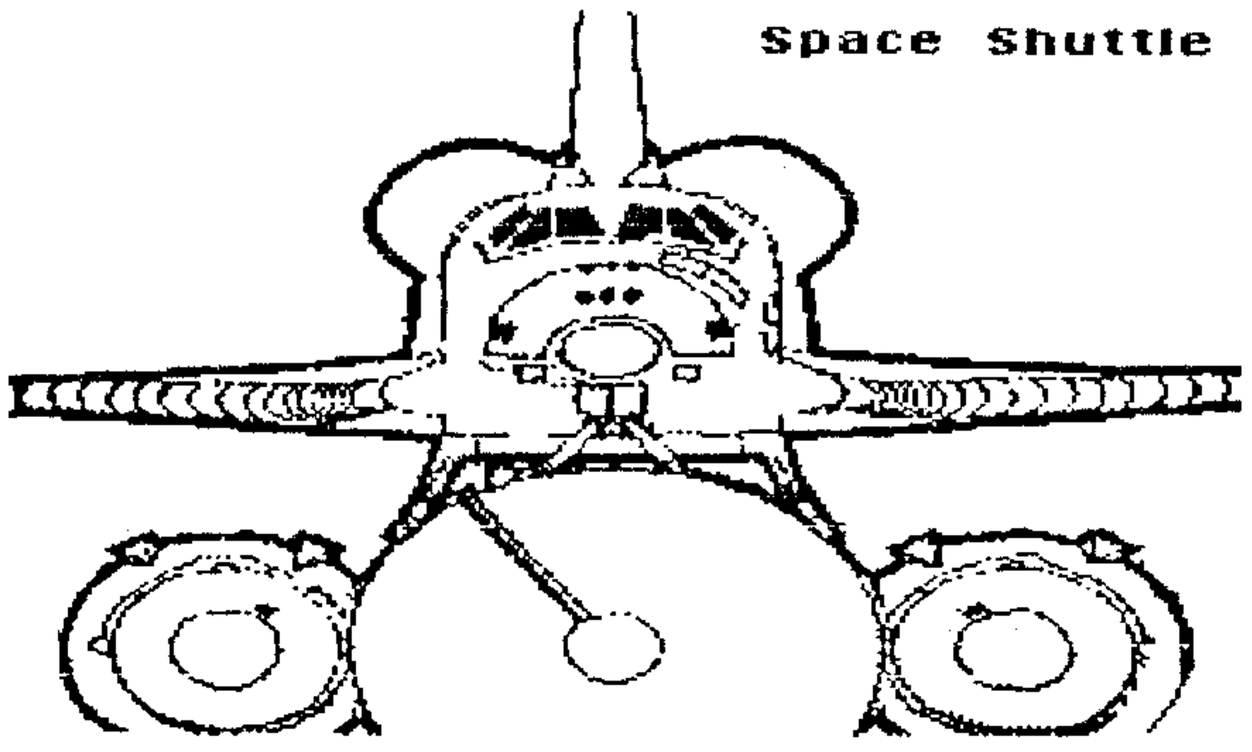
[similar to shorter program in FUNCPLUS!]



SPACELAB

```
100 CALL CLEAR :: CALL SCREE
N(6)
110 REM
120 REM ********************
        PGM BY SAM MOORE JR
        SHERMAN, TX 9/27/81
        ********************
130 REM (from LA Topics)
140 A$="<SPACE GEM>" :: FOR
GG=1 TO 7 :: DISPLAY AT(RND*
20,RND*20)BEEP:A$ :: NEXT GG
150 PRINT "DIRECTIONS? <Y/N>
"
160 CALL KEY(0,K,S)
170 IF S=0 THEN 160
180 IF K<>89 THEN 270
190 PRINT : :"THE OBJECT IS
TO MANEUVER   YOUR SPACE SHI
P TO AVOID    BEING HIT BY T
HE OTHER      SPACESHIPS."
200 PRINT :"A RUNNING TOTAL
IS KEPT OF  THE NUMBER OF TI
MES YOU ARE OVERRUN. THE OBJ
ECT, OF     COURSE IS TO MAK
E IT THROUGH"
210 PRINT "UNSCATHED. TO MAN
EUVER-ENTERS OR D OR E OR X
(ARROWS)."
220 PRINT : :"THE COMPUTER W
ILL ASK YOU   WHAT VELOCITY
YOU WANT.":"<2> IS A GOOD ST
ART."
230 PRINT : :"PRESS ANY KEY
TO CONTINUE..."
240 CALL KEY(0,K,S)
250 IF S=0 THEN 240
260 CALL CLEAR
270 PRINT "WHAT IS THE VELOC
ITY OF YOUR"
280 PRINT "SPACESHIP?(1-9)"
290 CALL KEY(0,K,S)
300 IF S=0 THEN 290
310 CALL CLEAR
320 V=K-48
330 V=V*10
340 PRINT "SKILL LEVEL DETER
MINES HOW  LONG THE GAME WIL
L RUN AND  SPEED OF THE ENEM
Y.":""
350 PRINT "WHAT SKILL LEVEL?
(1-9)"
360 CALL KEY(0,K,S)
370 IF S=0 THEN 360
380 LVL=K-48
390 CALL CLEAR :: CALL SCREE
N(4)
400 REM SPACE GEM
410 REM DEFINE SPACESHIPS
420 A$="0000070F107F7F10"
430 B$="0000E0F008FEFE08"
440 C$="0F070B112060F0F0"
450 D$="F0E0D08804060F0F"
460 CALL CHAR(104,A$)
470 CALL CHAR(106,B$)
480 CALL CHAR(105,C$)
490 CALL CHAR(107,D$)
500 CALL MAGNIFY(4)
510 REM MAKE SPACESHIPS
520 CALL SPRITE(#1,104,9,125
,100)
530 FOR AA=10 TO 15
540 SPEED=RND*LVL/5*60+RND*2
0
550 CALL SPRITE(#AA,104,16,1
,AA*45-445,SPEED,0):: NEXT A
A
560 CALL SCREEN(2)
570 REM MOVE RED SHIP
580 CALL KEY(0,K,S)
590 IF K<>68 THEN 600 :: CAL
L MOTION(#1,0,V):: GOTO 650
600 IF K<>83 THEN 610 :: CAL
L MOTION(#1,0,-V):: GOTO 650
610 IF K<>69 THEN 620 :: CAL
L MOTION(#1,-V,0):: GOTO 650
620 IF K<>88 THEN 630 :: CAL
L MOTION(#1,V,0):: GOTO 650
630 CALL MOTION(#1,0,0)
640 REM CHECK FOR HIT
650 CALL COINC(ALL,CC)
660 IF CC THEN 720
670 KK=KK+1
680 IF KK>29 THEN 810
690 MM=MM+1
700 IF MM=60+LVL*40 THEN 760
710 GOTO 560
720 CALL SCREEN(9)
730 HIT=HIT+1
740 FOR ZZ=1 TO 4 :: CALL SO
UND(-400,-5,5,ZZ*11+110,9,ZZ
*12+110,9):: NEXT ZZ
750 GOTO 560
760 REM END OF GAME
770 CALL SCREEN(4):: PRINT "
END OF GAME": :"YOU SUFFERED
";HITS;"HITS"
780 PRINT : : : : : : :
790 FOR D=1 TO 999 :: NEXT D
800 END
810 REM CHANGE ENEMY MOTION
820 KK=KK-28
830 FOR AA=10 TO 15 :: SPEED
=RND*LVL/9+10
840 CALL SPRITE(#AA,104,16,1
,AA-455,SPEED,0)
850 NEXT AA :: GOTO 700
```





Space Shuttle

DEBUGGING

by Jim Peterson

When you have finished *writing a program, the next* thing you should do is to run it. And, very probably, it will crash!

Don't be discouraged. It happens to the very best of programmers, very often.

So, the next thing to do is to debug it. And you are lucky that you are using a computer that helps you to debug better than some that cost ten times as much.

There are really three types of bugs. The first type will prevent the program from running at all - *it will crash with an error* message. The second type will allow the program to run, but will give the wrong results.

And the third type, which is not really a bug but might be mistaken for one, results from trying to run a perfectly good program with the wrong hardware, or with faulty hardware. As for instance, trying to run a Basic program, which uses character sets 15 and 16, in Extended Basic.

First, let's consider the *first type. The smart little* TI computer makes three separate checks to be sure your program is correct. First, when you key in a program line and hit the Enter key, it looks to see if there is anything it can't understand - such as a misspelled command or an unmatched quotation mark. If so, it will tell you so, most likely by SYNTAX ERROR, and refuse to accept the line.

Next, when you tell it to RUN the program, it first takes a quick look through the entire program, to find any combination of commands that it will not be able to perform. This is when it may crash with an error message telling you, for instance, that you have a NEXT without a matching FOR, or vice versa.

And finally, while it is actually running and comes to something that it just can't do, it will crash and give you an error message - *probably because a variable* has been given a value that cannot be used, such as a CALL HCHAR(R,C,32) when R happens to equal 0.

The TI has a wide variety of error messages to tell you when you did something wrong, what you did wrong, and where you did it wrong. But, it can be fooled! For instance, try to enter this program line (note the missing quotation mark).

```
100 PRINT "Program must be s
aved in:"merge format."
```

And, sometimes you may be told that you have a STRING-NUMBER MISMATCH when there is no string involved, because the computer has tried to read a garbled statement as a string.

Also, the line number given in the error message is the line where the computer found it impossible to run the program; that line may actually be correct but the variables at that point may contain bad values due to an error in some previous line.

If the error occurs in a program line which consists of several statements, and you cannot spot the error, you may have to break the line into individual single-statement lines. This is the easiest way to do that - Be sure the line numbers are sequenced far enough apart. Bring the problem line to the screen, put a ! just before the first ::, and enter *it. Bring it back to the* screen with FCTN 8, retype the line number 1 higher, use FCTN 1 to delete the first statement and the ! and ::, put a ! before the first ::, and continue. Then, when you have solved the bug, just delete the ! from the original line and delete all the temporary lines.

Pages 212-215 of your Extended Basic manual list almost all the error codes, and almost all the causes of each one - it will pay you *to consult these pages* rather than guessing what is wrong.

You may create some really bad bugs when you try to modify a program that was written by someone else - especially if you add any new variable names or CALLs to the program. Your new variable might be one that is already being used in the program for something else, perhaps in a subscripted array. I have noticed that programmers rarely use @ in a variable name, so I always tack it onto the end of any variable that I add to a program.

Also, the program that you are modifying may have ON ERROR routines, or a prescan, already built in. The ON ERROR routine was intended to take care of a different problem than the one you create, so it could lead you far astray - you had better delete that ON ERROR statement until you *are through modifying.*

The prescan had better be the subject of another lesson, but if the program has an odd-looking command !@P- up near the front somewhere, it has a prescan built in. And if so, if you add a new variable name or use a CALL that isn't in the program, you will get a SYNTAX ERROR even though there is no error. One way to solve this is to insert a line with !@P+ just before the problem line, and another with !@P- right after it.

When a program runs, even though it crashes or is stopped by FCTN 4 or a BREAK, the values assigned by the program to variables up to that point will remain in memory until you RUN again, or make a change to the program, or clear the

memory with NEW. This can be very useful. For instance, if the program crashes with BAD VALUE IN 680, and you bring line 680 to the screen and find it reads
CALL HCHAR(R,C,CH)
just type PRINT R;C;CH and you will get the values of R, C and CH at the time of the crash. You will find that R is less than 1 or more than 24, or C is less than 1 or more than 32, or CH is out of range.

In Extended Basic, you can even enter and run a multi-statement line in immediate code (that is, without a line number), if no reference is made to a line number. So, you can dump the current contents of an array to the screen by
FOR J=1 TO 100::PRINT A(J);:: NEXT J - or you can even open a disk file or a printer to dump it to.

You can also test a program by assigning a value to a variable from the immediate mode. If you BREAK a program, enter A=100 and then enter CON, the program will continue from where it stopped but A will have a value of 100.

You can temporarily stop a program at any time with FCTN 4, of course (the manual says SHIFT C, but it *was written for the old* 99/4), and restart it from that point with CON. Or you can insert a temporary line at any point, such as 971 BREAK if you want a break after line 970. Or, you can put a line at the beginning of the program listing the line numbers before which you want breaks to occur, such as 1 BREAK 960,970,980 Note that in this case the program breaks just BEFORE those listed line numbers. You can also use BREAK followed by one or more line numbers as a command in the immediate mode.

The problem with using BREAK and CON is that BREAK upsets your screen display format, resets redefined characters and colors to the default, and deletes sprites. So, it is sometimes better to trace the assignment of values to your variables by adding a temporary line to DISPLAY AT their values on some unused part of the screen. If you want to trace them through several statements, it will be better to GOSUB to a DISPLAY AT. And if you need to slow up the resulting display, just add a CALL KEY routine to the subroutine.

Sometimes, your program will appear to be not flowing through the sequence of lines you intended (perhaps because it dropped out of an IF statement to the next line!) and you will want to trace the line number flow. This can be done with TRACE, either as a command from the *immediate mode* or as a program statement, which will cause each line number to print to the screen as it is executed. If used as a command, it will trace everything from the beginning of the program, so it is usually better to insert a temporary line with TRACE at the point where you really want to start. Once you have implemented TRACE, the only way to get rid of it is with UNTRACE.

TRACE has its limitations because it can't tell you what is going on within a multi-statement line, and it will certainly mess up any screen display. Sometimes it is better to insert temporary program lines to display line numbers. I use CALL TRACE( ) with the line number between the parentheses, and a subprogram after everything else
30000 SUB TRACE(X)::DISPLAY AT(24,1):X :: SUBEND

Some programmers use ON ERROR combined with CALL ERR as a debugging tool, but I can't tell you much about that because I have never used it. ON ERROR can give more trouble than help if not used very carefully, and I cannot see that CALL ERR gives any information not available by other means.

Sometimes you can debug a line by simply retyping it. It is only very rarely that the computer is actually interpreting a line differently than it appears on the screen, but retyping may result in correcting a typo error that you just could not see. In fact, most bugs turn out to be very simple errors.

When you are debugging a string-handling routine, don't take it for granted that a string is really as it appears on the screen - it may have invisible characters at one or both ends. Try PRINT LEN(M$) to see if it contains more characters than are showing; or
PRINT "*"&M$&"*" to see if any blanks appear between the asterisks and the string.

There is no standard way to debug a program. Each problem presents a challenge to figure out what is going wrong, to devise a test to find out what is really happening.

Don't debug by experimenting, by changing variable values just to see what will happen, etc. Even if you succeed, you will not have learned what was wrong so you will not have learned anything - and if your program contains lines that you didn't understand when you wrote them, you will have real problems if you ever try to modify the program. (Believe me, I speak from experience!)

DAYTONA 99er - MAY 87

TOOLS AND TIPS

Q U I E T P. E. B.

Does your expansion box sound like the Concorde about to take off? Here's a simple cure that will not only quite it down but give increased cooling too!

Locate a sprite style fan such as the Torin TA3008. Be sure it is the 3.125" square model or it won't fit. Also, be sure it requires 115VAC. Used fans can be found cleaned and tested for about $ 10. For about $ 15.00 you can buy a new fan, although I've found the used fans to be just as dependable.

You will need to completely disassemble the expansion box to get to the fan but all that is needed is a Phillips screw-driver a small nutdriver. Remove the old fan, splice in the two power wires, and bolt your new fan on the same bolt studs. Re-assemble the box and viola, you now have a super quite system!

Fans can vary in air noise, but one of my systems is now about 1/2 as loud as the original and the other system is so quite that I have to look at the lights to be sure it's on. Yet both systems now move considerably more air.

>>>

Want a sharper display with your black white TV? Add this line at the start of your program:

```
CALL SCREEN(15)
```

This will disable the color generating circuit in the computer and remove the vertical lines often seen on BW TV's.

It also increases the sharpness of the characters.

Here's a short routine to clear the screen instead of using CALL CLEAR in your Extended Basic program:

```
J=33 :: K=0 :: FOR I=1 TO 16 :: J=J-1
:: K=K+1  :: CALL VCHAR(1,J,32,24) ::
NEXT I
```

Give it a try and see how it pop up your program. It gives a curtain effect with the clearing starting at the screen edges and moving toward the center. To see how it works, type a line before it:

```
CALL HCHAR(1,1,42,768)
```

-Bill Knoch
HUG TIBBS
06/17/85

***** PEB POWER SUPPLY *****

Here is a tip that may save you some frustration if you ever have to work on the power supply in the peripherial expansion box.

The primary of the transformer is fused as one would expect. However, the secondary is fused also. You can look for it as long as you like but you won't find it unless... you break loose the plastic surrounding the transformer. Inside, you'll find a fuse soldered to the secondary leads!!

I think it's great that they fused the secondary --- but like this???

David Anderso

Tools and tips subaitted by ED PRINCE

CHRISTMAS CARDS

Description

This program records addresses for Christmas cards and letters and identifies those that have been sent or received.

Functions of the Program

The program reads names, addresses, and sent/received indicators and then prints a formatted listing for use in addressing and mailing your Christmas cards.

Instructions for Use

Enter your Christmas card list prior to running the program. Maintain the data entries for use in determining next year's mailing requirements, adding and deleting from the list as necessary.

Data Entry

All data is entered by means of DATA statements.

Data Format

The data is formatted as follows:

Name, Street address, City, State, Zip, Sent, Received

Output Description

See example provided.

Comments

You may not wish to use the sent/received portion of the program. This function can be eliminated easily from the program (check lines 140, 150, and 220-260).

```
10 CALL CLEAR
20 REM  CHRISTMAS CARD LIST PROGRAM BASIC
30 REM  ********************* DATA INITIALIZATION AREA **********
40 M=1000
50 S=1
60 PRINT
70 REM  ******************** PROCESSING AREA *********************
80 PRINT
90 PRINT "  SENT RCVD"
100 FOR I=1 TO M
110 A1$=" "
120 A2$=" "
130 READ N$
140 IF N$="Y" THEN 220
150 IF N$="N" THEN 230
160 IF S<>1 THEN 165 ELSE 170
165 GOSUB 300
170 IF N$="END" THEN 380
180 S=0
190 N1$=N$
```

from TI BASIC COMPUTER PROGRAMS FOR THE HOME by Charles D. Sternberg

```
210 GOTO 280
220 A1$=N$
230 READ N$
240 S=0
250 IF N$<>"Y" THEN 270
260 A2$=N$
270 GOTO 130
280 NEXT I
290 REM  ******************** PRINT ROUTINE ***********************
300 PRINT " ("|A1$|") ("|A2$|")"|TAB(10)|N1$|" "
310 PRINT TAB(9)|S1$
320 PRINT TAB(4)|S2$|" "|S3$|" "|Z$
330 PRINT
340 K=K+1
350 S=1
360 RETURN
370 REM  ****************** PROGRAM TERMINATION POINT ************
380 PRINT
390 PRINT
400 PRINT
410 PRINT "NUMBER OF ENTRIES- "|K
420 PRINT
430 STOP
440 REM  ****************** DATA ENTRIES FOLLOW ******************
450 DATA JIM ANY NAME,111 ANY STREET,ANOTHER TOWN,STATE,11111
460 DATA Y,N
470 DATA JUDY DOE,234 MAIN STREET,HOME TOWN,USA,02341
480 DATA N,N
490 DATA TOM AND JERRY FORREST,44 WILLOW LANE,LUMBERTOWN,ALASKA,09876
500 DATA N,Y
510 DATA END
```

```
>RUN

  SENT RCVD
 (Y)  (  ) JIM ANY NAME
        111 ANY STREET
   ANOTHER TOWN STATE 11111

 (  ) (  ) JUDY DOE
        234 MAIN STREET
   HOME TOWN USA 02341

 (  ) (Y)
TOM AND JERRY FORREST
        44 WILLOW LANE
   LUMBERTOWN ALASKA 09876
NUMBER OF ENTRIES-  3
```

MAJOR SYMBOL TABLE - CHRISTMAS CARDS

NAME	.. DESCRIPTION
M	.. MAXIMUM NUMBER OF DATA READS
N$	.. NAME/SENT-RCVD INDICATOR
N1$	.. NAME
S1$	.. STREET ADDRESS
S2$	.. CITY ADDRESS
S3$	.. STATE ADDRESS
Z$	.. ZIP CODE
A1$	.. CARD SENT INDICATOR FOR PRINTING
A2$	.. CARD RCVD INDICATOR FOR PRINTING

FUNCTIONS USED

NAME	.. DESCRIPTION
TAB	.. FORMATS PRINT LINES