COMPUTE! Interviews Wendy Carlos And Frank Zappa

COMPUTE!

$2.95
January
1986
Issue 68
Vol. 8, No. 1
$3.75 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

SpeedCalc
A Powerful Spreadsheet
Program Inside For Commodore 64 And 128

Introduction To AmigaDOS
Getting Started With
The Most Powerful DOS Ever

***SpeedScript* Fontmaker**
For Commodore 64 & 128
Use Your Own
Custom Character Sets

Atari RESET Controller
Trapping The RESET Key
In BASIC Programs

Apple Keyboard Customizer
Rearrange Your Keyboard
With This Simple Program

Advanced IBM
Function Key Techniques
Add Extra Flexibility

BLAIR

71486 02193

COMPUTE!

JANUARY 1986
VOLUME 8
NUMBER 1
ISSUE 68

NOTE: See page 120 before typing in programs.

AP Apple, **Mac** Macintosh, **AT** Atari, **ST,** Atari ST, **V** VIC-20, **64** Commodore 64, **+4** Commodore Plus/4, **16** Commodore 16, **128** Commodore 128, **P** PET/CBM, **TI** Texas Instruments, **PC** IBM PC, **PCjr** IBM PCjr, **AM** Amiga, *General interest

COMPUTE! Publications, Inc.
One of the ABC Publishing Companies.
ABC Publishing, President, Robert G. Burton
1330 Avenue of the Americas, New York, New York 10019
Address all inquiries to:
P.O. Box 5406, Greensboro, NC 27403

COMPUTE! The Journal for Progressive Computing (USPS: 537250) is published monthly by COMPUTE! Publications, Inc., P.O. Box 5406, Greensboro, NC 27403 USA. Phone: (919) 275 9809. Editorial Offices are located at 324 West Wendover Avenue, Greensboro, NC 27408. Domestic Subscriptions: 12 issues, $24. POSTMASTER: Send address changes to: **COMPUTE!** Magazine, P.O. Box 10955, Des Moines, IA 50950. Second class postage paid at Greensboro, NC 27403 and additional mailing offices. ISSN 0194-357X.

copies of PLUG program library disks by sending a self-addressed, postage-paid mailer and blank, formatted disk(s) to PLUG. Members are encouraged to submit Plus/4 programs to the PLUG program library as well. However, all programs must be in the public domain or used with the author's express permission (PLUG does not condone piracy). Please direct all inquiries to:

PLUG
The Plus/4 Users' Group
Box 1001
Monterey, CA 93940

Thanks for the information. Plus/4 users should also note that the Programmer's Reference Guide for the Commodore Plus/4 *will soon become available. Though we haven't seen a copy at the time of this writing, the publisher (Scott, Foresman & Co.) has scheduled the book for release in late 1985 or early 1986, with a projected list price of $19.95.*

TI-99/4A Subprograms

What advantage, if any, does a TI Extended BASIC subprogram (with SUB) have over an ordinary subroutine called with GOSUB?

Dan Schwarz

An Extended BASIC subprogram is quite similar to a subroutine. Both contain a series of BASIC statements set apart from the main program and are often used to perform a repetitive function. However, while a routine accessed with GOSUB may appear anywhere in the program, a subprogram must appear at the end. Subprograms are also called by name rather than by line number. For instance, the following statement calls a subprogram named MULT:

```
10 CALL MULT(A,B,C)
```

Inside the parentheses is the parameter list *or set of variable values you want to pass from the main program to the subprogram. This is necessary because the system treats the subprogram as a separate program: The subprogram can't recognize any variables used in the main program unless you pass their values in the parameter list. Here's how the subprogram MULT would begin:*

```
500 SUB MULT(D,E,F)
```

The SUB statement marks this as a subprogram. Since MULT expects to receive three variables from the main program, its parameter list contains three items. Parameters are passed according to their position within the parameter list: That is, the first parameter in the subprogram's list becomes equal to the first one in the calling list. In this case, D equals A, E equals B, and F equals C. You can use any simple variable names, of course. All subprogram variables are local, *meaning they have no effect on the main program. In this case, you can use D, E, and F in the subprogram any way you like without affecting main program variables of the same name.*

The end of a subprogram is marked with this statement:

```
580 SUBEND
```

One reason to use subprograms is that the computer can find them much faster than ordinary subroutines. During the prescan *phase of program execution, the computer looks at the entire program text, noting (among other things) the location of any subprograms. When the subprogram is CALLed, the computer already knows its location and begins executing its statements without delay. To locate an ordinary subroutine, on the other hand, the computer must scan the entire program for the right line number, which takes significantly longer.*

Because subprograms are called by name rather than line number, their placement in the main program is not dependent on line numbers. This feature, plus the use of local variables, means you can build up a library of program modules. Whenever you need one of the subprograms, you can easily merge it into the program you're working on. Since subprograms are always placed at the end of the main code, the program tends to be more structured and easier to understand. Similar features are common in more structured programming languages.

Resetting The 1541 Drive

Please publish a reset circuit I can add to my 1541 disk drive. I understand that resetting the drive by turning the power off and on is not particularly good for it.

Robert Desko

As with computers, there are two ways to reset the 1541 disk drive. A hardware reset is triggered physically (grounding the microprocessor's RESET line). A software reset is activated by a command that makes the device perform its normal powerup routines without actually switching the power off and on. Software resets save a little stress on the chips. For instance, SYS 64738 on the Commodore 64 has much the same effect as turning the power on. The equivalent command for the 1541 is UJ, as shown in this program:

```
10 OPEN 15,8,15
20 GOSUB 70
30 PRINT#15,"UJ"
40 FOR J=1 TO 1000:NEXT
50 GOSUB 70
60 CLOSE 15:END
70 INPUT#15, ER, ER$, TR, SE
80 PRINT ER;ER$;TR;SE
90 RETURN
```

This program displays the drive status twice, before and after the drive is reset. Here's what you'll see on the screen:

```
0 OK 0 0
73 CBM DOS V2.6 1541 0 0
```

*The first message indicates normal (no error) status. Though it's transmitted like an error message, the second message doesn't indicate an error. It's a "signature" which the drive generates every time you turn it on (like the 64's familiar powerup message *** COMMODORE 64 BASIC V2 ***). Once the command channel is open (line 10), you can reset the drive at any time by sending the characters UJ with a PRINT# command (line 30). You may replace the J in UJ with a colon, although there's no practical advantage in doing so.*

The delay loop in line 40 is needed because it takes the drive a moment or two to clear its internal memory buffers, set zero page variables, and complete other reset tasks. During that brief interval the drive can't respond to any other commands. To reset the drive from direct mode (when you're not running a program), type OPEN 15,8,15,"UJ" and press RETURN. Wait a second or two, then enter CLOSE 15 to close the command channel.

In most circumstances, a software reset is as effective as a hardware reset and has the advantage of resetting the drive without disturbing anything in the computer's memory. If you can't bring the drive back with UJ or by pressing RUN/STOP–RESTORE, you must do a hardware reset. Since the 1541 uses a 6502 microprocessor, building a reset switch is no more difficult than building one for the 64 or VIC-20. All you need is a momentary-contact, normally open switch wired between the 6502's reset line (RESET) and its ground line (GND). Since these lines are available on pins 6 (RESET) and 2 (GND) of the 6-pin DIN connector at the back of the drive, it's possible to make a switch that plugs directly into the serial port connector. Your disk drive manual contains a diagram of the pins. It's a good idea to debounce the switch by wiring a small capacitor in parallel with the switch terminals. Use extreme caution *when attempting this modification: If you don't understand exactly how to build the switch, get help from a friend who does or refer the work to a qualified technician.*

There's one disadvantage to performing a hardware reset. Since the serial cable connects to the RESET line in the computer's microprocessor, pressing a reset switch on the drive resets the computer as well—destroying any BASIC program in memory. Grounding RESET anywhere on the serial bus resets every serial device in the system. If you have a reset switch on your computer, an expansion card, etc., you'll rarely need a separate switch for the drive.

SOLITAIRE

Ben Elizer

Looking for something different to do with your computer? Like the conventional game of solitaire, this computerized version requires you to think ahead at all times. The original program runs on the Commodore 64, Plus 4, 16, and 128, and we've added versions for Apple II-series computers, the IBM PC with color/graphics adapter and BASICA, IBM PCjr with Cartridge BASIC, TI-99/4A, and Atari 400/800, XL, and XE with at least 16K RAM.

"Solitaire" is an electronic version of the familiar card game. Like the original, this game challenges you to put a deck of cards in order using the fewest possible moves. Type in and save the program listed for your computer, then read the instructions before you play the game.

Unshuffling The Deck

As you probably know, Solitaire has a very simple object. After shuffling a deck of playing cards, you must put them back in order, following a few simple rules. Though there are several different variations of the conventional game, here are the rules for this version:

When you run the program, the computer deals out four rows of 13 cards, then removes the aces, leaving four empty spaces. Your goal is to rearrange the cards into four rows of the same suit, putting the cards in each row in ascending order from the lowest (2) to the highest (king), without leaving any empty spaces between cards. That sounds simple enough. But since you must move a card into one of the four empty spaces, your choices for any given move are limited.

Your position on the screen is shown by a blinking cursor. Press the M key to move from the current position to another empty space. When you press P, the computer moves a card into the current space: Which card it puts there depends on which card is immediately to the left of the space. Whenever possible, the computer uses the next card in suit. For example, if the card to the left of your current position is the 2 of hearts, pressing P puts the 3 of hearts in the current space and puts a space where the 3 of hearts was before. If you press P on a space to the right of the queen of diamonds, the king of diamonds moves from its current position to that space, and so on. Each time you press P, one space is filled and another is emptied.

In this way you can gradually move cards into the right order. When you press P on a space at the beginning of a row, the computer asks which suit to play (hearts, clubs, spades, or diamonds). This determines the suit for that row. While it's possible to win on only one deal, most games require two or more deals. When no moves are possible (every empty space is followed by a king or another space), the computer automatically shuffles the remaining cards and deals them out again. Of course, it does not disturb cards that are already in correct order. You'll find that it takes considerable foresight to win consistently in only two or three deals. Completely random play results in an average of nine or ten deals.

Commodore Versions

Program 1 is Solitaire for the Commodore 64. It also works as listed for the Commodore 128. For the Commodore 16 and Plus/4, change line 20 to read as follows:

```
20 COLOR0,2,5:COLOR4,7,0
```

Apple Version

This version of Solitaire is in two parts. Program 4 is the main BASIC

To keep yourself from being burned to a crisp by a fire-breathing dragon, just cast a *nitfol* spell. Oops. Make that a *gondar* spell.

One moment you're slipping the first disk of The Enchanter Trilogy in your computer—the next you're a novice magician, chosen by fate to battle an evil that threatens the very foundations of magic. And since you're the main character, every decision you make will determine the story's outcome. Suppose, for instance, that when faced with a giant fire-breathing dragon you decide to cast a *nitfol* spell, which allows you to converse with beasts in their own tongue. You simply type, in plain English:

```
>CAST THE NITFOL SPELL ON THE DRAGON
```

And the story responds:

```
THE DRAGON PAUSES AND ROARS OUT
A BENEVOLENT GREETING, WHICH, TO
YOUR CHAGRIN, FRIES YOU TO A
DELICATE CRISP. YOU HAVE DIED.
```

Suppose, on the other hand, you decide to invoke a spell that quenches open flames:

```
>CAST THE GUNDAR SPELL ON THE
DRAGON
```

In that case the story responds:

```
THE DRAGON'S FLAME IS
DOUSED IN A TORRENT OF
RUSHING WATER. IT
DISAPPEARS WITH A
TORTUOUS SCREAM.
```

As you progress from each story in this spellbinding trilogy to the next, the dangers you must face, the spells you must learn and the puzzles you must unravel grow to thrilling proportions. But so will your powers. Conquer the evil that lurks within all three and you'll be elevated from sorcerer's apprentice to Leader of the Circle of Enchanters.

And along the way you'll face situations that are as lively as the text that describes them:

```
YOU FEEL VERY FUNNY, SORT OF SQUASHED AND
PUSHED AND SQUEEZED. YOUR SURROUNDINGS ARE
WAVERING, THEN GROWING, THEN WAVERING AGAIN.
THE FEELING VANISHES, BUT YOUR SURROUNDINGS
ARE TEN TIMES THEIR FORMER SIZE...OR IS IT
THAT YOU ARE ONE-TENTH YOUR FORMER SIZE?
```

Infocom addicts think The Enchanter Trilogy is our best work ever. So *aimfiz* yourself into your local software store today. And let it cast its spell on you.

Progress from Standard Level ENCHANTER® to new Expert Level SPELLBREAKER™ and you'll gain the powers of a true Sorcerer.

INFOCOM™

For more information call 1-800-262-6868. Or write to us at 125 CambridgePark Dr., Cambridge, MA 02140

Available for most personal computers.

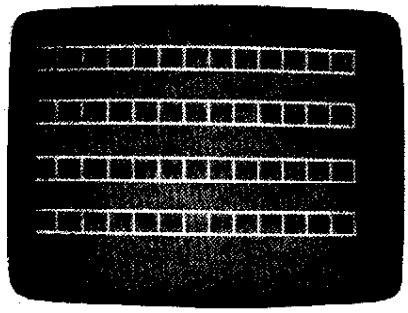
program and Program 5 is a special binary file used to create the high resolution card displays. You must type in Program 5 with "Apple MLX," COMPUTE!'s machine language entry program for Apple, published elsewhere in this issue. Follow the MLX instructions carefully and be sure to save a copy of the program when you are done. Here are the addresses you need for MLX:

Starting address: 8000
Ending address: 8317

The program works on any Apple II-series computer, but the graphics look much better on a color monitor.

TI-99/4A Version

This version (Program 6) plays exactly like the others except that the rows of cards are displayed vertically rather than horizontally.



"Solitaire" for the Commodore 64.

Program 1: Commodore Solitaire

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" published bimonthly in COMPUTE!.

```
10 CLR:A=RND(-TI):PRINT"{CLR}"
   ;:F5=1:C=52:W=1          :rem 21
20 POKE53281,15:POKE53280,14
                            :rem 34
30 DIMP(4,13),D(52),T(52)
                            :rem 2
40 GOSUB920:REM--DRAW BOXES--
                            :rem 0
50 PRINT"{HOME}{BLU}";TAB(15);
   "SHUFFLING..."           :rem 224
60 GOSUB1020:REM--SHUFFLE DECK
   --                       :rem 159
70 PRINT"{HOME}";TAB(15);"DEAL
   ING...{6 SPACES}"        :rem 17
80 FORI=1TO4:FORJ=1TO13:GOSUB1
   080:NEXTJ:NEXTI          :rem 197
90 GOSUB1210:REM--FIND FIRST F
   OUR EMPTY BOXES          :rem 26
100 PRINT"{HOME}";"{BLU}TYPE '
    M' TO MOVE TO THE NEXT EMP
    TY SLOT"                :rem 94
110 PRINT"OR 'P' TO PLACE A CA
    RD AT THE CURSOR"       :rem 89
120 IFW=5THENW=1:F1=0:GOSUB128
    0:IFF1=0THEN570         :rem 250
130 I=INT((E(W)-1)/13)+1:J=E(W
    )-13*INT((E(W)-1)/13)
                            :rem 181
140 IFJ=1THEN170            :rem 163
150 LL=P(I,J-1):IFLL/13=INT(LL
    /13)THENW=W+1:GOTO120
                            :rem 61
160 IFLL=1ORLL=14ORLL=27ORLL=4
    0THENW=W+1:GOTO120      :rem 60
170 N$="W":S$=" ":PRINT"{BLU}"
    ;:GOSUB1160             :rem 123
180 GETX$:IFX$=""THENPRINT"
    {YEL}";:GOSUB1160       :rem 22
190 IFX$<>""THEN210         :rem 37
200 GETX$:IFX$=""THENPRINT"
    {BLU}";:GOSUB1160:GOTO180
                            :rem 156
210 IFX$="P"THEN240         :rem 52
220 IFX$="M"THEN:N$=" ":GOSUB1
    160:W=W+1:GOTO120       :rem 210
230 GOTO180                 :rem 103
240 IFJ=1THEN370            :rem 166
250 LL=P(I,J-1)             :rem 42
260 IFLL/13=INT(LL/13)THEN180
                            :rem 190
270 IFLL=1ORLL=14ORLL=27ORLL=4
    0THEN190                :rem 138
280 TE=P(I,J):TT=T(P(I,J)):L=T
    (P(I,J-1)+1)            :rem 118
290 T(P(I,J))=T(P(I,J-1)+1)
                            :rem 156
300 T(P(I,J-1)+1)=TT        :rem 55
310 P(I,J)=P(INT((L-1)/13+1),L
    -13*INT((L-1)/13)):rem 245
320 P(INT((L-1)/13)+1,L-13*INT
    ((L-1)/13))=TE          :rem 47
330 GOSUB1080               :rem 223
340 I=INT((L-1)/13)+1:J=L-13*I
    NT((L-1)/13):GOSUB1080
                            :rem 88
350 GOSUB1210:W=1:GOTO120
                            :rem 225
360 REM--OFFER CHOICE OF 'TWOS
    '                       :rem 36
370 PRINT"{HOME}":PRINT"
    {19 DOWN}"              :rem 204
380 PRINT"{BLU}";           :rem 198
390 PRINT"NOW YOU HAVE A CHOIC
    E OF"                   :rem 3
400 PRINT"WHICH '2' YOU WANT T
    O PLACE"                :rem 151
410 PRINT"TWO OF 'S','H','D',O
    R 'C'";                 :rem 175
420 GETT$:IFT$=""THEN420
                            :rem 117
430 IFT$="S"THENN2=2:GOTO490
                            :rem 160
440 IFT$="H"THENN2=15:GOTO490
                            :rem 202
450 IFT$="D"THENN2=28:GOTO490
                            :rem 203
460 IFT$="C"THENN2=41:GOTO490
                            :rem 198
470 GOTO420                 :rem 106
480 REM--NOW EXCHANGE LOCATION
    S--                     :rem 23
490 TE=P(I,J):TT=T(P(I,J)):L=T
    (N2)                    :rem 223
500 T(P(I,J))=T(N2)         :rem 252
510 T(N2)=TT                :rem 160
520 PRINT"{HOME}":PRINT"
    {19 DOWN}"              :rem 201
530 PRINT"{25 SPACES}":rem 105
540 PRINT"{30 SPACES}":rem 106
550 PRINT"{30 SPACES}";
                            :rem 166
560 GOTO310                 :rem 104
570 FORI=1TO4               :rem 17
580 N(I)=0                  :rem 242
590 IFP(I,1)<>2ANDP(I,1)<>15AN
    DP(I,1)<>28ANDP(I,1)<>41TH
    EN650                   :rem 219
600 N(I)=1                  :rem 236
610 FORJ=2TO12              :rem 61
620 IFP(I,J)-1<>P(I,J-1)THENJ=
    14:GOTO640              :rem 69
630 N(I)=N(I)+1             :rem 2
640 NEXTJ                   :rem 35
650 NEXTI                   :rem 35
660 IFN(1)=12ANDN(2)=12ANDN(3)
    =12ANDN(4)=12THEN1350
                            :rem 98
670 F5=F5+1                 :rem 44
680 REM--ERASE THE WRONG ENTRI
    ES                      :rem 212
690 PRINT"{HOME}{39 SPACES}"
                            :rem 131
700 PRINT"{39 SPACES}";
                            :rem 163
710 N$=" ":S$=" "           :rem 189
720 PRINT"{HOME}{BLU}";TAB(15)
    ;"RESHUFFLING..."       :rem 171
730 FORI=1TO52:D(I)=I:NEXT
                            :rem 89
740 FORI=1TO4               :rem 16
750 FORJ=N(I)+1TO13         :rem 85
760 GOSUB1160               :rem 229
770 NEXT:NEXT               :rem 86
780 C3=52                   :rem 185
790 FORI=1TO4               :rem 21
800 IFN(I)=0THEN820         :rem 69
810 FORJ=1TON(I):D(P(I,J))=0:N
    EXTJ                    :rem 34
820 NEXTI                   :rem 34
830 FORI=1TO4:FORJ=1+N(I)TO13
                            :rem 3
840 R1=INT(RND(1)*C3+1)
                            :rem 250
850 IFD(R1)=0THEND(R1)=D(C3):C
    3=C3-1:GOTO840          :rem 16
860 P(I,J)=D(R1)            :rem 83
870 D(R1)=D(C3):C3=C3-1:NEXTJ
                            :rem 131
880 NEXTI                   :rem 40
890 FORII=1TO52:T(II)=0:NEXT
                            :rem 233
900 GOSUB1060               :rem 224
910 GOTO70                  :rem 58
920 REM--SET UP BOXES       :rem 235
930 PRINT"{WHT}";           :rem 173
940 FORI=1TO4               :rem 18
950 PRINT:PRINT             :rem 242
960 FORJ=1TO12:PRINT"**{R}";:N
    EXT:PRINT"**{S}"        :rem 110
970 FORJ=1TO13:PRINT"
    {2 SPACES}-";:NEXT:PRINT
                            :rem 169
980 FORJ=1TO12:PRINT"**{E}";:N
    EXT:PRINT"**{X}"        :rem 126
990 NEXT                    :rem 225
1000 PRINT"{HOME}";         :rem 224
1010 RETURN                 :rem 162
1020 REM--SET UP DECK       :rem 169
1030 FORI=1TO52:D(I)=I:NEXT
                            :rem 131
1040 FORI=1TO4:FORJ=1TO13:R1=I
     NT(RND(1)*C+1):P(I,J)=D(R
     1):D(R1)=D(C):C=C-1
                            :rem 48
1050 NEXT:NEXT              :rem 126
1060 FORII=1TO4:FORJJ=1TO13:T(
     P(II,JJ))=(II-1)*13+JJ:NE
     XT:NEXT                :rem 62
1070 RETURN                 :rem 168
1080 REM--SHOW CARD P(I,J)
                            :rem 194
1090 S$="ASZX":S$=MID$(S$,INT(
     (P(I,J)-1)/13)+1,1)
                            :rem 127
1100 PRINT"{BLK}";:IFS$="S"ORS
     $="Z"THENPRINT"{RED}";
                            :rem 188
1110 N=P(I,J)-13*INT((P(I,J)-1
     )/13)                  :rem 71
```

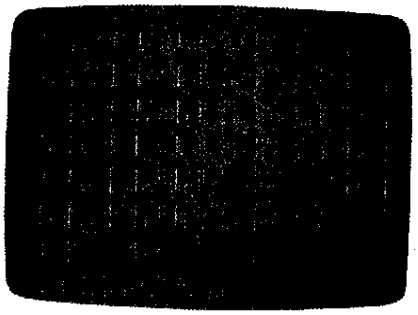
```
10 2010 PRINT "YOU WON!!": PRINT
       "IT TOOK YOU ";F5;" TRI
       ES"
70 2020 PRINT "TYPE 'Y' TO PLAY
       AGAIN";
92 2030 OCT X$: IF ASC (X$) > 96
       THEN X$ = CHR$ ( ASC (X
       $) - 32)
73 2040 IF X$ = "N" THEN END
E6 2050 IF X$ = "Y" THEN RUN
78 2060 GOTO 2030
```

Program 5: Apple Graphics File

For instructions on entering this listing, please refer to the "Apple MLX" article published elsewhere in this issue.

```
START ADDRESS:  8000
END ADDRESS:    8313

8000: 20 EC 80 B0 72 20 02 B2 02
8009: D0 6D 20 5D 02 D0 60 AD CD
8010: 20 83 F0 0F A9 7F 8D 26 9D
8018: 83 8D 27 83 A9 07 8D 28 08
8020: 83 D0 0F A9 55 8D 26 83 44
8028: A9 2A 8D 27 83 A9 05 8D 07
8030: 28 83 A9 03 8D 14 83 A9 F8
8038: 02 8D 1C 83 20 0B 81 A9 33
8040: 18 8D 15 83 20 9A 81 20 1A
8048: 5E 81 EE 18 83 CE 15 83 3D
8050: D0 F2 AD 20 83 F0 20 A0 EE
8058: 00 AD 21 83 20 88 80 A0 E5
8060: 04 AD 21 83 20 88 80 A0 EF
8068: 08 AD 20 83 20 88 80 A0 D9
8070: 0C AD 20 83 20 88 80 60 A3
8078: 74 82 03 08 94 82 0A 08 FF
8080: AC 82 01 01 AC 82 0A 0F 3B
8088: 0A 0A 0A 8D 25 83 B9 78 4E
8090: 80 85 FC C8 B9 78 80 85 95
8098: FD C8 AD 22 83 8D 1A 83 AC
80A0: B9 78 80 C8 18 6D 23 83 79
80A8: C9 07 90 07 E9 07 EE 1A 36
80B0: 83 B0 F5 8D 1B 83 B9 78 0A
80B8: 80 18 6D 24 83 8D 18 83 F5
80C0: A9 01 8D 14 83 A9 07 8D 28
80C8: 1C 83 A9 08 8D 13 83 AC E2
80D0: 25 83 B1 FC 8D 26 83 20 77
80D8: 0B 81 20 9A 81 20 5E 81 38
80E0: EE 18 83 EE 25 83 CE 15 A8
80E8: 83 D0 E4 60 20 69 82 C9 F7
80F0: 35 90 01 60 A2 00 8E 21 2A
80F8: 83 C9 0E 90 07 E9 0D EE E1
8100: 21 83 B0 F5 E9 00 8D 20 74
8108: 83 18 60 AD 14 83 8D 16 99
8110: 83 A8 A9 00 99 26 83 AD 4E
8118: 1B 83 18 6D 1C 83 C9 07 EC
8120: 90 05 E9 07 EE 16 83 8D BE
8128: 1D 83 AD 26 83 09 7F 8D 7F
8130: 17 83 AC 1B 83 F0 15 A2 93
8138: 00 0E 26 83 BD 26 83 0A 53
8140: 3E 27 83 E8 EC 16 83 D0 C2
8148: F3 88 D0 EB AC 16 83 B9 BE
8150: 26 83 09 80 2D 17 83 99 D6
8158: 26 83 88 10 F2 60 AC 1D F0
8160: 83 B9 8C 81 AC 1A 83 00 8A
8168: 31 FE 19 26 83 91 FE 88 32
8170: 30 0A F0 08 B9 26 83 91 AB
8178: FE 88 D0 F8 AC 1B 83 B9 59
8180: 93 81 A0 00 31 FE 0D 26 87
8188: 83 91 FE 60 7F 7E 7C 78 FE
8190: 70 60 40 00 01 03 07 0F 1D
8198: 1F 3F AD 18 83 29 3F A8 1A
81A0: B9 C2 81 05 E6 85 FF AD AC
81A8: 18 83 29 08 F0 02 A9 80 A1
81B0: 18 2C 18 83 70 04 10 04 BD
81B8: 69 28 69 28 6D 1A 83 85 8A
81C0: FE 60 00 04 08 0C 10 14 40
81C8: 18 1C 00 04 08 0C 10 14 C3
81D0: 18 1C 01 05 09 0D 11 15 0B
81D8: 19 1D 01 05 09 0D 11 15 D3
81E0: 19 1D 02 06 0A 0E 12 16 1B
81E8: 1A 1E 02 06 0A 0E 12 16 E3
81F0: 1A 1E 03 07 0B 0F 13 17 2B
81F8: 1B 1F 03 07 0B 0F 13 17 F3
8200: 1B 1F A9 00 8D 1A 83 8D FA
8208: 1B 83 20 69 82 8D 19 83 16
8210: C0 01 90 12 F0 01 60 C9 FE
8218: 18 90 01 60 A9 24 8D 1A 86
8220: 83 A9 04 8D 1B 83 A9 00 E4
8228: 8D 1F 83 A9 E0 8D 1E 83 C3
8230: AD 19 83 CD 1E 83 90 04 C3
8238: ED 1E 83 38 2E 1F 83 4E F2
8240: 1E 83 90 EF 18 6D 1B 83 76
8248: 8D 1B 83 8D 23 83 18 AD 29
8250: 1F 83 6D 1A 83 8D 1A 83 1F
8258: 8D 22 83 18 60 20 69 82 77
8260: 8D 18 83 8D 24 83 C9 C0 FE
8268: 60 20 B1 00 20 05 E1 A5 5A
8270: A1 A4 A0 60 7F 7F 5F 07 49
8278: 01 01 0F 7F 7F 7F 77 55 56
8280: 55 57 5F 7F 7F 7F 5F 57 F9
8288: 55 57 5F 7F 7F 7F 47 47 C1
8290: 01 01 47 7F 7F 7F 7F 7E AE
8298: 78 78 7F 7F 7F 7F 7F 7A 51
82A0: 7A 7E 7F 7F 7F 7F 7F 7E E1
82A8: 7A 7E 7F 7F 7F 7F 7F 7F EA
82B0: 7E 7E 7F 7F 63 1C 1F 1F 65
82B8: 63 7C 7C 00 63 1C 1F 63 4B
82C0: 1F 1F 1C 63 4F 47 43 49 3E
82C8: 4C 00 4F 4F 00 7C 7C 60 1E
82D0: 1F 1F 1C 63 43 79 7C 60 40
82D8: 1C 1C 1C 63 00 1C 1F 4F AA
82E0: 67 73 73 73 63 1C 1C 63 43
82E8: 1C 1C 1C 63 63 1C 1C 1C 9C
82F0: 03 1F 1C 63 63 1C 1C 1C D8
82F8: 1C 1C 1C 63 1F 1F 1F 1F 9F
8300: 1F 1F 1C 63 63 1C 1C 1C F7
8308: 1C 10 44 13 1C 1C 1C 60 C4
8310: 1C 1C 1C 1C FF FF 00 00 71
```



"Solitaire" for TI-99/4A computers.

Program 6: TI-99/4A Solitaire

Version by Patrick Parrish, Programming Supervisor

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" published bimonthly in COMPUTE!.

```
100 DIM P(4,13),D(52),T(52)
110 CALL SCREEN(16)
120 GOTO 170
130 FOR I7=1 TO LEN(H$)
140 CALL HCHAR(ROW,COL+I7,A
    SC(SEG$(H$,I7,1)))
150 NEXT I7
160 RETURN
170 CALL CLEAR
180 PRINT TAB(10);"SOLITAIR
    E":::::::::::
190 GOSUB 2780
200 F5=1
210 C=52
220 W=1
230 RANDOMIZE
240 GOSUB 1760
250 H$="...SHUFFLING"
260 ROW=24
270 COL=9
280 GOSUB 130
290 GOSUB 1920
300 CALL HCHAR(24,10,32,12)
310 H$="...DEALING"
320 GOSUB 130
330 FOR I=1 TO 4
340 FOR J=1 TO 13
350 GOSUB 2110
360 NEXT J
370 NEXT I
380 CALL HCHAR(24,10,32,10)
390 GOSUB 2300
400 GOSUB 3010
410 IF W<>5 THEN 470
420 W=1
430 F1=0
440 GOSUB 2450
450 IF F1<>0 THEN 470
460 GOTO 1200
470 I=INT((E(W)-1)/13)+1
480 J=E(W)-13*INT((E(W)-1)/
    13)
490 IF J=1 THEN 540
500 LL=P(I,J-1)
510 IF (LL<>1)*(LL<>14)*(LL
    <>27)*(LL<>40)*(LL/13<>
    INT(LL/13))THEN 540
520 W=W+1
530 GOTO 410
540 N$="v"
550 S$=" "
560 GOSUB 2250
570 CALL KEY(0,KK,SS)
580 IF SS<>0 THEN 660
590 N$=" "
600 GOSUB 2250
610 CALL KEY(0,KK,SS)
620 IF SS<>0 THEN 660
630 N$="v"
640 GOSUB 2250
650 GOTO 570
660 IF KK=80 THEN 720
670 IF KK<>77 THEN 570
680 N$=" "
690 GOSUB 2250
700 W=W+1
710 GOTO 410
720 IF J=1 THEN 900
730 LL=P(I,J-1)
740 IF (LL/13=INT(LL/13))+(
    LL=1)+(LL=14)+(LL=27)+(
    LL=40)THEN 570
750 TE=P(I,J)
760 TT=T(P(I,J))
770 L=T(P(I,J-1)+1)
780 T(P(I,J))=T(P(I,J-1)+1)
790 T(P(I,J-1)+1)=TT
800 P(I,J)=P(INT((L-1)/13+1
    ),L-13*INT((L-1)/13))
810 P(INT((L-1)/13)+1,L-13*
    INT((L-1)/13))=TE
820 GOSUB 2110
830 I=INT((L-1)/13)+1
840 J=L-13*INT((L-1)/13)
850 GOSUB 2110
860 GOSUB 2300
870 W=1
880 GOTO 410
890 REM  OFFER CHOICE OF TW
    O'S
900 CALL HCHAR(23,1,32,64)
910 H$="WHICH '2' YOU WANT
    TO PLACE?"
920 ROW=23
930 COL=1
940 GOSUB 130
950 H$="TWO OF 'S','H','D',
    OR 'C' ?"
960 ROW=24
970 GOSUB 130
980 CALL KEY(0,KK,SS)
990 IF KK<>83 THEN 1020
1000 N2=2
1010 GOTO 1110
1020 IF KK<>72 THEN 1050
1030 N2=15
```

```
1040 GOTO 1110
1050 IF KK<>68 THEN 1080
1060 N2=28
1070 GOTO 1110
1080 IF KK<>67 THEN 980
1090 N2=41
1100 REM  EXCHANGE LOCATION
     S
1110 TE=P(I,J)
1120 TT=T(P(I,J))
1130 L=T(N2)
1140 T(P(I,J))=T(N2)
1150 T(N2)=TT
1160 CALL HCHAR(23,1,32,64)
1170 GOSUB 3010
1180 GOTO 800
1190 REM  PLAYER CAN NO LON
     GER MOVE SO ERASE THE
     WRONG ENTRIES, RESHUFF
     LE, & DEAL
1200 FOR I=1 TO 4
1210 N(I)=0
1220 IF (P(I,1)<>2)*(P(I,1)
     <>15)*(P(I,1)<>28)*(P(
     I,1)<>41)THEN 1300
1230 N(I)=1
1240 FOR J=2 TO 12
1250 IF P(I,J)-1=P(I,J-1)TH
     EN 1280
1260 J=14
1270 GOTO 1290
1280 N(I)=N(I)+1
1290 NEXT J
1300 NEXT I
1310 IF (N(1)=12)*(N(2)=12)
     *(N(3)=12)*(N(4)=12)TH
     EN 2560
1320 F5=F5+1
1330 REM  ERASE THE WRONG E
     NTRIES
1340 CALL HCHAR(23,1,32,60)
1350 N$=" "
1360 S$=" "
1370 H$="...RESHUFFLING"
1380 ROW=24
1390 COL=9
1400 GOSUB 130
1410 FOR I=1 TO 52
1420 D(I)=I
1430 NEXT I
1440 FOR I=1 TO 4
1450 FOR J=N(I)+1 TO 13
1460 GOSUB 2250
1470 NEXT J
1480 NEXT I
1490 C3=52
1500 FOR I=1 TO 4
1510 IF N(I)=0 THEN 1550
1520 FOR J=1 TO N(I)
1530 D(P(I,J))=0
1540 NEXT J
1550 NEXT I
1560 FOR I=1 TO 4
1570 FOR J=1+N(I)TO 13
1580 RANDOMIZE
1590 R1=INT(RND*C3+1)
1600 IF D(R1)<>0 THEN 1640
1610 D(R1)=D(C3)
1620 C3=C3-1
1630 GOTO 1590
1640 P(I,J)=D(R1)
1650 D(R1)=D(C3)
1660 C3=C3-1
1670 NEXT J
1680 NEXT I
1690 FOR II=1 TO 52
1700 T(II)=0
1710 NEXT II
1720 GOSUB 2040
1730 CALL HCHAR(24,10,32,14
     )
1740 GOTO 310
1750 REM  DRAW BOXES
1760 PRINT " ** **  ** **
     ** **  ** **"
1770 FOR J=1 TO 6
1780 FOR I=1 TO 2
1790 PRINT "a  e  ba  e  ba
       e  ba  e  b"
1800 NEXT I
1810 PRINT " dd dd  dd dd
     dd dd  dd dd"
1820 NEXT J
1830 PRINT :::::
1840 FOR I=0 TO 3
1850 CALL HCHAR(19,7+I*7,99
     ,2)
1860 CALL VCHAR(20,3+I*7,97
     ,2)
1870 CALL HCHAR(22,4+I*7,99
     ,2)
1880 CALL VCHAR(20,6+I*7,98
     ,2)
1890 NEXT I
1900 RETURN
1910 REM  SET UP DECK
1920 FOR I=1 TO 52
1930 D(I)=I
1940 NEXT I
1950 FOR I=1 TO 4
1960 FOR J=1 TO 13
1970 RANDOMIZE
1980 R1=INT(RND*C+1)
1990 P(I,J)=D(R1)
2000 D(R1)=D(C)
2010 C=C-1
2020 NEXT J
2030 NEXT I
2040 FOR II=1 TO 4
2050 FOR JJ=1 TO 13
2060 T(P(II,JJ))=(II-1)*13+
     JJ
2070 NEXT JJ
2080 NEXT II
2090 RETURN
2100 REM  SHOW CARD P(I,J)
2110 S$="ytux"
2120 H5=INT((P(I,J)-1)/13)+
     1
2130 S$=SEG$(S$,H5,1)
2140 N8=P(I,J)-(H5-1)*13
2150 IF N8<>1 THEN 2170
2160 S$=" "
2170 IF (H5=1)+(H5=4)THEN 2
     200
2180 N1$=" hijklmnopqrs"
2190 GOTO 2210
2200 N1$=" 23456789JQK"
2210 N$=SEG$(N1$,N8,1)
2220 GOSUB 2250
2230 RETURN
2240 REM  PLACE N$:S$ AT PO
     SITION I,J
2250 J5=J+(J>7)*7
2260 CALL HCHAR(J5*3,(I-1)*
     7+4-(J>7)*3,ASC(N$))
2270 CALL HCHAR(J5*3-1,(I-1
     )*7+5-(J>7)*3,ASC(S$))
2280 RETURN
2290 REM  FIND FIRST FOUR E
     MPTY BOXES
2300 Z=1
2310 FOR I=1 TO 52 STEP 13
2320 E(Z)=T(I)
2330 Z=Z+1
2340 NEXT I
2350 FOR J=1 TO 4
2360 FOR I=1 TO 3
2370 IF E(I)<=E(I+1)THEN 24
     10
2380 AA=E(I)
2390 E(I)=E(I+1)
2400 E(I+1)=AA
2410 NEXT I
2420 NEXT J
2430 RETURN
2440 REM  CHECK TO SEE IF A
     LL FOUR SPACES FOLLOW
     A KING OR BLANK
2450 FOR K=1 TO 4
2460 X=INT((E(K)-1)/13+1)
2470 Y=E(K)-13*INT((E(K)-1)
     /13)
2480 IF Y<>1 THEN 2500
2490 F1=1
2500 W2=P(X,Y-1)
2510 IF (W2=1)+(W2=14)+(W2=
     27)+(W2=40)+(W2/13=INT
     (W2/13))THEN 2530
2520 F1=1
2530 NEXT K
2540 RETURN
2550 REM  ALL DONE
2560 H$="CONGRATULATIONS!!
     YOU WON!!"
2570 CALL HCHAR(23,1,32,64)
2580 ROW=23
2590 COL=2
2600 GOSUB 130
2610 H$="IT TOOK YOU "&STR$
     (F5)&" TRIES."
2620 ROW=24
2630 COL=5
2640 GOSUB 130
2650 CALL HCHAR(23,1,32,32)
2660 H$="PLAY AGAIN (Y/N)?"
2670 ROW=23
2680 COL=8
2690 GOSUB 130
2700 CALL KEY(0,KK,SS)
2710 IF SS=0 THEN 2700
2720 IF KK<>89 THEN 2750
2730 CALL CLEAR
2740 GOTO 200
2750 IF KK<>78 THEN 2700
2760 END
2770 REM  REDEFINE CHARS
2780 FOR I=96 TO 101
2790 READ A$
2800 CALL CHAR(I,A$)
2810 NEXT I
2820 DATA 00000000000000FF,
     0101010101010101,8080808
     0808080808080
2830 DATA FF0000000000000,
     FF0000000000FF,8181818
     1818181818181
2840 FOR I=104 TO 118
2850 READ A$
2860 CALL CHAR(I,A$)
2870 NEXT I
2880 DATA 003844404810207C,
     003844441804443C,00001
     828487C0808
2890 DATA 007C407804044438,
     0018204078444438,007C4
     4081020202020
2900 DATA 003844443844443C,
     003844443C040830,00384
     44444444444438
2910 DATA 0004040404044438,
     0038444444544834,00444
     85060504844
2920 DATA 00367F7F3E1C0800,
     00183C7E7E3C1800,FFFFF
     FFFFFFFFFFF
2930 CALL COLOR(10,7,1)
2940 CALL COLOR(11,7,1)
2950 FOR I=120 TO 121
2960 READ A$
2970 CALL CHAR(I,A$)
2980 NEXT I
2990 DATA 001C1C7777081C00,
     00183C7E7E183C00
3000 RETURN
3010 H$="<M>OVE TO NEXT EMP
     TY SLOT"
3020 ROW=23
3030 COL=3
3040 GOSUB 130
3050 H$="<P>LACE A CARD AT
     CURSOR"
3060 ROW=24
3070 GOSUB 130
3080 RETURN
```

The Beginners Page

Tom R. Halfhill, Editor

The Power Of Strings

Last issue we introduced the concept of string variables and briefly hinted at their power—that their ability to hold strings of characters can let your programs manipulate words and sentences instead of just numbers. Consider for a moment how many programs manipulate text in some way: text editors, word processors, database managers, telecommunications programs, educational software, adventure games, even spreadsheets to some extent. Because math isn't the only language humans use to communicate ideas and manipulate information, over the years we've devised ways to make computers handle our alphabets as well.

But keep in mind that digital computers are still number-crunchers at heart. The alphabetic characters which appear on their monitor screens are merely an illusion created for our convenience. Internally, computers see the whole universe in terms of numbers, and they're unaware of anything that can't be translated into numbers. We'll discover some implications of this as we explore the uses of strings in BASIC.

Reducing Redundancy

Probably the simplest way to begin taking advantage of strings in your programs is to use them to save memory and reduce typing. When you assign a string of characters to a string variable (A$="HELLO"), the computer stores the string in a safe place in memory. The string variable is like a bookmarker that reminds the computer where it is keeping the string. From then on, whenever you include that string variable in a BASIC statement, the computer looks up the string of characters in memory and carries out your command. If you print the variable, the entire string appears on the screen.

For example, if there are screen messages that frequently appear in different parts of your program—such as "PRESS ANY KEY TO CONTINUE" or "SELECT NUMBER OF MENU CHOICE"—it's a waste of memory and time to repeatedly type them in as separate PRINT statements. Instead, assign them to string variables like this:

```
10 A$="PRESS ANY KEY TO
   CONTINUE"
20 B$="SELECT NUMBER OF MENU
   CHOICE"
```

and then print the appropriate variable when you need to display the message:

```
100 PRINT A$
```

Here's another example: You've probably seen programs which draw horizontal rows of asterisks or dashes across the screen to make decorative borders, or to separate the screen into different sections for menus and so forth. Obviously it would waste memory to draw these lines with literal PRINT statements, since each PRINT would have to be followed by 40 or 80 characters (depending on the width of your computer's screen display). A better way is to use a FOR-NEXT loop, such as FOR X=1 TO 40:PRINT "*";:NEXT X. But if your program draws these lines often, you might save even more memory by defining a string variable with asterisks or dashes and then just printing the variable whenever you need it. This also executes faster than a FOR-NEXT loop.

Strings With INPUT

Substituting string variables for literal PRINT statements is useful, but you really begin appreciating the power of string variables when you use them as *variables*. Like numeric variables, string variables can be manipulated in dozens of ways.

For instance, with an INPUT statement you can allow the user to assign and reassign characters to a string variable as the program runs—something a literal string can never do. Here's the most common example:

```
10 PRINT "WHAT IS YOUR NAME";
20 INPUT N$
30 PRINT "HELLO, ";N$
40 GOTO 10
```

(Make sure you type the semicolons *outside* the quotation marks in lines 10 and 30, and include the space between the comma and closing quotes in line 30. On Atari computers, don't forget you must always dimension a string variable before its first reference—insert the statement DIM N$(50) with a line number less than 10.)

When you run this program, it prints the message in line 10 and then waits at line 20 until the user types some characters and presses RETURN or ENTER. When the computer detects that RETURN or ENTER is pressed, it assigns whatever characters were typed to the string variable N$. Then it continues to line 30 and prints the HELLO message followed by the characters in N$. Finally, the computer returns to line 10 and lets the user assign a completely new string of characters to N$.

Since the content of N$ is determined by the user, not predetermined by the programmer, this little program can be the basis for a branching routine which takes different actions depending on the user's response. And that, in turn, is the basis for a wide variety of programs which tailor themselves to user input: educational programs that ask a question and evaluate the answer, programs that offer options and accept yes or no choices, programs that request you to specify a filename before loading or saving a data file—just about every kind of program, in fact. We'll take a closer look at these techniques and others in next month's column. ©

Programming the TI

C. Regena

Music And Sound On The TI

Music and sound on the TI can be a lot of fun and fairly easy to program. Some computers require several statements to even play one note, but the TI can play an entire chord with one statement. The best way to learn to program music and sound is to sit at the console and experiment. This month we'll look at a few techniques.

The basic sound statement is CALL SOUND(d,f,v) where d is duration, f is frequency, and v is volume. You may specify more than one frequency and volume for each statement to hear more voices.

The duration parameter tells the computer how many milliseconds (thousandths of a second) the sound should last. CALL SOUND (1000,262,1) plays middle C for exactly one second. You can use this feature for any kind of timing, with or without sound. For example, by setting the volume to the softest and using a high frequency out of hearing range, a program can silently count off seconds.

In music programs it's helpful to use a variable for the duration. For example, let T represent a quarter note. T/2 will be an eighth note, T/3 a triplet, 2*T a half note, 4*T a whole note, and so on. Before the sound statements, define a value for T.

```
110 T=400
120 CALL SOUND(T,262,2)
130 CALL SOUND(T/2,294,2)
140 CALL SOUND(T/2,330,2)
150 CALL SOUND(2*T,349,2)
160 CALL SOUND(4*T,392,2)
170 END
```

To change the tempo, you won't need to change each sound statement, only line 110. For example, change set T=200, then RUN. The tempo changes with all the notes in proportion.

The TI can execute other statements, such as calculations or graphics, while making sounds. Last month's Christmas program is an example of graphics commands being executed among music commands. If another sound statement is encountered, the computer waits until the previous duration is finished. If you want the computer to execute a sound statement without waiting for the previous duration to finish, use a negative number for the duration:

```
110 CALL SOUND(2000,440,2)
120 CALL SOUND(-400,262,2)
130 END
```

The first note should be played for two seconds. However, line 120 includes a negative duration, so its sound starts as soon as the computer gets to line 120, and the sound continues for 400 milliseconds. Negative durations are often placed in a FOR-NEXT loop:

```
110 FOR F=262 TO 392 STEP 12
120 CALL SOUND(-200,F,2)
130 NEXT F
140 END
```

To determine frequency values for notes, consult the charts in the manuals that came with your computer. You can use these charts to translate sheet music. For example, CALL SOUND(1000,440,2) plays A at concert pitch. To play a chord, you can list three frequencies and volumes with one duration in a statement:

```
CALL SOUND(1200,262,2,330,2,392,2)
```

But you're not limited to numbers on the chart. For example, the frequency for middle C is 262, and the frequency for D is 294. You can play any tone between these notes:

```
110 FOR F=262 to 294
120 CALL SOUND(300,F,2)
130 PRINT F
140 NEXT F
150 END
```

By varying the frequency in a FOR-NEXT loop, you can create interesting sound effects:

```
110 FOR F=440 TO 523 STEP 15
120 CALL SOUND(-100,F,2)
130 NEXT F
140 FOR F=262 TO 131 STEP -10
150 CALL SOUND(-100,F,2)
160 NEXT F
170 END
```

Create noises by using negative frequencies from -1 to -8. These noises can be fun to add to games. However, you're not limited to just these noises. You may combine up to three other frequencies with one noise—you can spend days experimenting with different combinations to make different noises. Try these examples:

```
CALL SOUND(1000,-6,2,440,2)
CALL SOUND(1000,-6,2,262,2)
CALL SOUND(1000,-6,2,131,2,165,2)
```

The volume parameter may be a value from 0 (loudest) to 30 (softest). You can assign different volumes to notes to create dynamics, such as a crescendo, or to make a melody more prominent.

```
110 CALL SOUND(400,262,8)
120 CALL SOUND(400,294,6)
130 CALL SOUND(400,330,4)
140 CALL SOUND(400,349,2)
150 CALL SOUND(800,392,0)
160 END
```

Try varying the volume in loops to create sound effects:

```
110 FOR V=0 TO 30
120 CALL SOUND(-100,262,V)
130 NEXT V
140 FOR V=30 TO 0 STEP -1
150 CALL SOUND( 100, 5,V)
160 NEXT V
170 FOR V=0 TO 30
180 CALL SOUND(-100,-6,V)
190 NEXT V
200 FOR F=262 TO 330 STEP 34
210 FOR V=0 TO 30
220 CALL SOUND(-100,F,V,-6,V)
230 NEXT V
240 NEXT F
250 END
```

CALL SOUND is quite versatile and can add a lot to your programs. Take the time to experiment and you'll discover that you can create all kinds of sounds with your TI. ©

Classified

SOFTWARE

THE INVESTMENT MANAGER a program for the 64 guaranteed to out-perform any other method. Comes with two programs to help your investment plan. All three $19.95 or write for free details to: Author's Club Software, 6027 S. High, Suite 410, Oklahoma City, OK 73149

TI-99/4A QUALITY SOFTWARE for Business, Home and Entertainment ** BONUS Software Offer! ** Send for FREE Catalog to MICRO-BIZ HAWAII, Box 1108 Pearl City, HI 96782

TI-99/4A Software/Hardware bargains. Hard-to-find items. Huge selection. Fast service. Free catalog. D.E.C., Box 690, Hicksville, NY 11801

BIBLE QUIZ GAMES and other Bible software for the C64. Fun and learn for all ages. BIBLE-MATCH-WITS 1 (easy), II (hard). STAIRWAY TO HEAVEN 1, 2, 3. Colorful, animation, graphics, and sound. Each $29.95. Order or send SASE for brochure to COMPEDS, P.O. Box 147, Narrows, VA 24124

STAR TRADER a program for 2 to 5 players. Show off your 64 when guests come over. Gomuko compiled basic program 15 by 15 board, Unmove, load/save game user changable logic $14.95 each. Author's Club Software, 6027 S. High, Suite 410, Oklahoma City, OK 73149

/SPEEDPAK/-C64 Speedscript 3.0 + enhancer. Adds alt. screens, macros, encryption, help screen, chr. set, mail merge & more! Disk. $15 to: /SPEEDPAK/ P.O.B. 22022, Dept. C1, Greensboro, NC 27420

COMPUQUEST ADVENTURES, Direct for less! 48K Atari (D), Apple (D), C64 (D or T), 48K Spectrum (T). New Special: "Transported" $12.95 + $3 s/h (WI res. 5% tax) to: COMPUQUEST, Box A492, St. Croix Falls, WI 54024. SASE for info.

MORSE CODE COURSE FOR COMMODORE 64 for the beginner as well as the AMATEUR who wants to improve speed. For info: F. Ardavin, 1254 Sweet Pine Dr., Norcross, Georgia 30093. Or send $19.95 for disk.

MEDICAL COMPUTER PROGRAMS in basic, $20/booklet, Alan Kwasman, MD, 2844 Taurus, Riverside, CA 92503 (714) 785-8957

TI-99/4A - 130 PROGRAMS $3 EACH! Catalog $1, refunded. Nuts & Bolts, diskfull of 100 mergeable utilities $19.95 ppd. Tigercub Tips, diskfull of 50 programs & C $15 ppd. TIGERCUB, 156 Collingwood, Columbus, OH 43213

FANTASTIC DAILY NUMBER FORECASTER! Guaranteed Winners or Money Back! Picks up to 3 STRAIGHT WINNERS most every week, playing 1 to 3 a day! Apple, IBM, C64, Atari, 1 drive. Many reports of hitting for THOUAND. Send SASE for info. $99.95 on disk only to: Z-Way, P.O. Box 9017, Canton, OH 44711

APPLE, ATARI AND COMMODORE OWNERS!!! DISCOUNT PRICES ON NAME BRAND SOFTWARE. SEND SASE FOR PRICE LISTING. SOFT WARE BROTHERS, POB 07095, MILW. WI 53207

TI-99/4A NEW STATES AND CAPITALS GAME Hi-Res map of USA. Send $12 for cass. Or $1 for more info. to: TRINITY SYSTEMS 1022 Grandview, Pittsburgh, PA 15237

PROGRAMS FOR THE TANDY 1000 Send $1 for list of educat. & entertain. progs. Refundable with first purchase. SODA POP SW, POB 653, Kenosha, WI 53141

MAILBOX - Fun, easy, efficient electronic mail for single station, multiuser C-64; passwords, printable output & more; also a great game for kids; disk only $7.95. Pete Kvale, 3156 Choctaw, Memphis, TN 38111

HARDWARE

IBM PC for Christmas? Build a compatible for a fraction of the cost. Easy to assemble. Write for free datalogue to JV Systems, P.O. Box 9807, Brook Park, OH 44142 or call (216) 842-4313

COMMODORE C128 80-CHARACTER CABLE. No need for RGB1 monitor. 80 char. on reg. monitor. Just $9.95 + $2 s/h to: UNITED RESEARCH, 7723 R'Horse Ln., Boerne, TX 78006

MISCELLANEOUS

64 AUTHOR'S CLUB - We get you published. Send for free demo-disk. Send $5.00 shipping and handling to: Author's Club, 6027 S. High, Suite 410, Oklahoma City, OK 73149

HELP IS ON THE WAY! Just call 1-800-334-0868 to get your free copy of the latest COMPUTE! Books Catalog! If you need help in getting information on all of the latest COMPUTE! book titles available plus all COMPUTE! backlist titles, call us today!

*** MR. SOFTWARE CO. ALL POPULAR TITLES *** * Printers, Monitors, Drives, VISA, MC * heavy discounts - Send $1.00 for catalog 11-9 Exton Complex, Somers Point, NJ 08244

HACKER'S HELPER holds magazine or book upright and keeps your place. It's like having three extra hands. Send $24.95 to John Keener, RD#3, Butler, PA 16001

C64 USERS - FREE BROCHURE! Game and instructional programs, each include detailed analysis, beg. or int. level. SASE to: C16 H.O.S, 19730 Ave 18, Madera, CA 93637

FREE CATALOG - TI-99, COMMODORE, IBM. SPECIFY. Hardware, Software, Accessories. Competition Computer, 2629 W. National, Milwaukee, WI 53204 (800) 662-9253

1541 PERMANENT ALIGNMENT $35. C64 repair $55, 1541 repair $85. DI-TECH, 701 East North Ave., #C Lompoc, CA 93436 (805) 736-9727

Maxell MD1, $1.29-MD2, $1.99. Dysan 104/1D, $1.79-104/2D, $2.39. Shipping $3.75. Also Verbatim, IBM, 3M, BASF. TAPE WORLD, 220 Spring St., Butler, PA 16001, 1-800-245-6000. Visa, MC.

DISK SALE! - SS/DD 35-trk for Apple w/sleeve & label-10/$5.80, bulk-100/$45. Standard SS/DD w/sleeve & label-10/$7.50, bulk-100/$59. DS/DD w/sleeve & label-10/$8.50, bulk-100/$67. 3½" SS for Mac-10/$19.99. PREMIUM QUALITY, LIFETIME WARRANTY! Money-back satisfaction guarantee! Min. order $20. Send check or pay by MC/VISA/AE $3 shipping, + $2 if C.O.D. - UNITECH, 20 Hurley St., Cambridge, MA 02141. (800) 343-0472, in Mass. (617) "UNI-TECH".

COMPUTE! Classified is a low-cost way to tell over 350,000 microcomputer owners about your product or service.

Rates: $25 per line, minimum of four lines. Any or all of the first line set in capital letters at no charge. Add $15 per line for boldface words, or $50 for the entire ad set in boldface (any number of lines.)

Terms: Prepayment is required. Check, money order, American Express, Visa, or MasterCard is accepted. Make checks payable to COMPUTE! Publications.

Form: Ads are subject to publisher's approval and must be either typed or legibly printed. One line equals 40 letters and spaces between words. Please underline words to be set in boldface.

General Information: Advertisers using post office box numbers in their ads must supply permanent address and telephone numbers. Orders will not be acknowledged. Ad will appear in next available issue after receipt.

Closing: 10th of the third month preceding cover date (e.g., June issue closes March 10th). Send order and remittance to: Harry Blair, Classified Manager, COMPUTE!, P.O. Box 5406, Greensboro, NC 27403. To place an ad by phone, call Harry Blair at (919) 275-9809.

Notice: COMPUTE! Publications cannot be responsible for offers or claims of advertisers, but will attempt to screen out misleading or questionable copy.