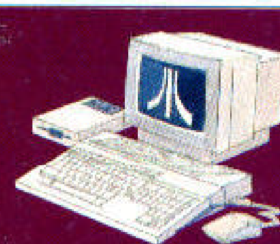
The Fall's New Games For Commodore, Apple, Atari, and IBM

COMPUTE!

$2.95
October
1985
Issue 65
Vol. 7, No. 10
$3.75 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

Atari 520 ST
A Hands-On Report



The Commodore 1541 Save-With-Replace Bug:
Proof That It's Real And How To Avoid It

The Witching Hour
Haunting Strategy Game
For Commodore 64, Atari, Apple, IBM PC/PCjr, TI-99/4A

Apple II
Pull-Down Menus
Add Mac-Style Features To Your II+, IIe, IIc

Lightning Renumber For Atari
Powerful Tool For BASIC Programmers

10
0 71486 02193

Earth will be destroyed in 12 minutes to make way for a hyperspace bypass.

Should you hitchhike into the next galaxy? Or stay and drink beer?

Simply slip the disk in your computer and suddenly you are Arthur Dent, the dubious hero of THE HITCHHIKER'S GUIDE TO THE GALAXY™ a side-splitting masterwork of interactive fiction by novelist Douglas Adams and Infocom's Steve Meretzky. And every decision you make will shape the story's outcome. Suppose for instance you decide to linger in the pub. You simply type, in plain English:

>DRINK THE BEER

And the story responds:

YOU GET DRUNK AND HAVE A TERRIFIC TIME FOR TWELVE MINUTES, ARE THE LIFE AND SOUL OF THE PUB, TELL SOME REALLY TERRIFIC STORIES, MAKE EVERYONE LAUGH A LOT, AND THEY ALL CLAP YOU ON THE BACK AND TELL YOU WHAT A GREAT CHAP YOU ARE AND THEN THE EARTH GETS UNEXPECTEDLY DEMOLISHED. YOU WAKE UP WITH A HANGOVER THAT LASTS FOR ALL ETERNITY. YOU HAVE DIED.

Suppose, on the other hand, you decide to:

>EXIT THE VILLAGE PUB THEN GO NORTH

In that case you'll be off on the most mind-bogglingly hilarious adventure any earthling ever had.

The Hitchhiker's Guide to the Galaxy comes complete with Peril Sensitive Sunglasses, a Microscopic Space Fleet, a DON'T PANIC Button, a package of Multipurpose Fluff and orders for the destruction of your home and planet.

You communicate – and the story responds – in full sentences. Which means that at every turn, you have literally thousands of alternatives. So if you decide it might be wise, for instance, to wrap a towel around your head, you just say so:

>WRAP THE TOWEL AROUND MY HEAD

And the story responds:

THE RAVENOUS BUGBLATTER BEAST OF TRAAL IS COMPLETELY BEWILDERED. IT IS SO DIM IT THINKS IF YOU CAN'T SEE IT, IT CAN'T SEE YOU.

But be careful about what you say. Or one moment you might be strapped down, forced to endure a reading of the third worst poetry in the galaxy; the next you could be hurtling through space with Marvin the Paranoid Android aboard a stolen spaceship.

And simply staying alive from one zany situation to the next will require every proton of puzzle solving prowess your mere mortal mind can muster. Even simple tasks can put you at wit's end:

>OPEN THE DOOR

And the story responds:

THE DOOR EXPLAINS, IN A HAUGHTY TONE, THAT THE ROOM IS OCCUPIED BY A SUPER-INTELLIGENT ROBOT AND THAT LESSER BEINGS (BY WHICH IT MEANS YOU) ARE NOT TO BE ADMITTED. "SHOW ME SOME TINY EXAMPLE OF YOUR INTELLIGENCE," IT SAYS, "AND MAYBE, JUST MAYBE I MIGHT RECONSIDER."

Other interactive science fiction stories from Infocom include PLANETFALL,™ in which you're stranded on a mysterious deserted world. STARCROSS,™ a puzzling challenge issued eons ago and light-years away. SUSPENDED,™ the race to stabilize an entire planet's life support systems. And A MIND FOREVER VOYAGING,™ a radically new work of serious science fiction in which you explore the future of mankind.

But don't panic. You'll be accompanied every light-year of the way by your trusty Hitchhiker's Guide, which you can always depend on for up-to-the-nanosecond information. Well, almost always:

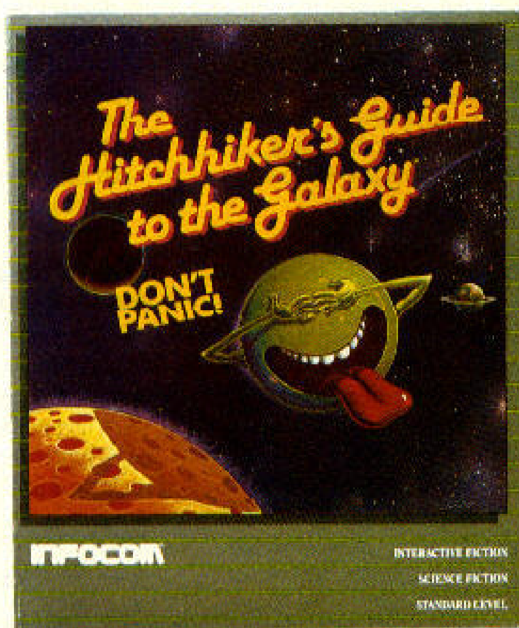
>CONSULT THE HITCHHIKER'S GUIDE ABOUT THE MOLECULAR HYPERWAVE PINCER

And the story responds:

SORRY, THAT PORTION OF OUR SUB-ETHA DATABASE WAS ACCIDENTALLY DELETED LAST NIGHT DURING A WILD OFFICE PARTY.

So put down that beer, take that towel off your head, open the door, hitchhike down to your local software store today and pick up THE HITCHHIKER'S GUIDE TO THE GALAXY. Before they put that bypass in.

Still not convinced? Try our Sampler Disk which includes portions of four different types of stories for a paltry $7.95. If it doesn't get you hooked on the addictive pleasures of Infocom, return it for a full refund. If it does, you can apply the price toward any Infocom story. You can't lose!



INFOCOM

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

COMPUTE!

OCTOBER 1985
VOLUME 7
NUMBER 10
ISSUE 65

AP Apple, **Mac** Macintosh, **AT** Atari, **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.
*General interest.

TOLL FREE Subscription Order Line
800-334-0868 (In NC 919-275-9809)

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.

Too Many Caesars

I own two Commodore computers and a 1541 disk drive. I would like to connect both computers to the drive at once (of course, I would only send disk commands from one computer at a time). Everything works fine when only one computer is turned on, but when I turn on the second one, the first computer does a cold start. When I try to send disk commands from either computer, the entire system seems to lock up. Is there any way to accomplish what I'm trying to do?

Charles Mitchell

Since you can connect more than one peripheral to a single computer, you might expect the reverse to be true. Why can't two computers share the same drive? The answer reveals a fundamental difference between a computer and peripheral devices such as disk drives and printers. The computer is designed to act as "absolute ruler" of the system. It not only sends and receives information (as peripherals can do), but also sends commands that control the whole system. Plugging two computers into the same disk drive is like creating a Rome with two Caesars: Each computer acts like the only command-giver in existence, and the system becomes confused.

In the first case you describe, turning on the second computer sends a normal reset command to every device in the system—including the second computer, which responds as if it had reset itself. Sending a disk command (which goes to the other computer as well as the drive) makes things even worse. Serial communications require a complex exchange of "handshaking" signals between computer and peripheral to make sure one doesn't send data until the other is ready, and vice versa. Since the second computer isn't designed to respond as a peripheral, it can't complete the handshake and crashes the entire system.

One makeshift way to do what you want is to unplug the serial cable from one computer whenever you want to use the other. However, we definitely don't recommend this as a regular practice. The serial port connectors aren't designed for such heavy use, and you run the risk of sending garbage signals along the line. For long-term use you may want to buy a switching box which cleanly disconnects one computer from the serial bus before connecting the other.

ACCEPT On TI

I have a problem using ACCEPT on my TI-99/4A with Extended BASIC. When I try to enter numeric input with ACCEPT and accidentally press ENTER before any input, the screen scrolls and I get an error message. Is there any way I can avoid this without using the CALL KEY statement?

Jory Rannow

The following program illustrates one solution to your problem:

```
100 CALL CLEAR
110 DISPLAY AT(1,1):"ROW
    #1"
120 ACCEPT AT(2,1)VALIDAT
    E(NUMERIC):X$
130 IF X$="" THEN 120
140 X=VAL(X$)
150 PRINT X
```

After this program clears the screen, line 110 prints a message on line 1 so you can tell whether scrolling occurs. Line 120 takes in numeric input (numerals 0–9, period symbol, plus symbol, minus symbol, or E) and accepts the input as X$. If at this point you hit ENTER by mistake, line 130 sends you back for another try without scrolling the screen. Once you've entered a value, line 140 converts it from a string into the numeric variable X.

Unwanted Commodore Messages

I have written a machine language routine that loads several program modules into the Commodore 64 from disk. However, the computer prints the usual SEARCHING FOR and LOADING messages during every load. How can I prevent these messages from appearing on the screen?

Allen Kotomski

These messages are generated by the 64's operating system, which controls input/output functions. Since Commodore calls the operating system the Kernal, they're known as Kernal control messages. One easy way to mask them is to change the character color to the same color as the screen background. The messages then print invisibly on the screen. However, since they may overprint an existing display or cause the screen to scroll, it's usually better to suppress them altogether.

Location $9D (157 decimal) holds a flag that tells the 64 what type of messages to display. When the flag contains 128 (bit 7 is set to 1), the computer prints Kernal control messages to tell you when it's searching, loading, saving, or verifying. When bit 7 is set to 0, control messages are not displayed. Though you rarely see them when using BASIC, the Kernal also has its own set of error messages. For instance, the Kernal equivalent of BASIC's FILE NOT FOUND message is I/O ERROR #4. Location $9D controls Kernal error messages as well: They're displayed when the flag contains 64 (bit 6 is set to 1), and suppressed when bit 6 is clear.

The Witching Hour

Brian Flynn

This game of skill and foresight is ideal for a bleak, stormy October night. Originally programmed for the IBM PC with color/graphics adapter and PCjr, versions have been added for the Commodore 64, expanded VIC-20, Atari 400/800, XL, and XE, TI-99/4A, and Apple II-series computers. The Commodore 64 and Atari versions require a joystick.

When autumn winds send a shiver down your spine and the witching hour draws near, there's no better entertainment than a good computer game. "The Witching Hour" is an absorbing contest of strategy based on Alquerque, a board game played in ancient Egypt and still popular in Spain today. Type in and save The Witching Hour, referring to the listing for your computer. Since every version is similar, read the general game rules below, then check the specific notes for your computer before running the program.

The Witching Hour pits broomstick-straddling witches against ethereal ghosts and is played on a board of 25 squares with 12 pieces to a side. After choosing sides, you attempt to take your opponent's players by jumping over them. You can move vertically, horizontally, or diagonally. However, certain diagonal moves are illegal (the lines between squares show where you can go) and only one square is vacant when the game begins.

Jumping an opposing player's

piece removes that piece from the board. If no capture is possible, you may move any piece to an adjacent empty square. You may not pass up a capture—if it's possible to jump an opponent, you must always do so—and if the first capture puts you in position to make another, you must jump again (except in the Apple version). The computer won't let you make illegal moves.

Play ends when all the pieces from one side have been removed from the board. You can play against a friend or measure your skills against the computer (the IBM and TI versions also let you watch the computer play itself). Like other contests of strategy,, The Witching Hour is simple to learn, but a challenge to master, and can be played at many different levels. Hint: It's sometimes smart to sacrifice a player to draw the opponent into a dangerous position.

IBM PC/PCjr Version

Each game square on the screen is marked with one of the letters of the alphabet. To move a piece, first type the letter for the square of the piece you want to move. Then type the letter of the square where you want to go. For instance, to move a witch from square L to square M, type L when the computer prompts you with FROM and type M when it prompts you with TO. If you press Enter without typing a letter, the computer takes that turn. Thus, to play alone against the computer, just press Enter every other turn. Press Enter on every turn to watch the computer play against itself.

Commodore 64 And VIC-20

Both Commodore versions of The Witching Hour offer a one- or two-player option when the game begins. The 64 version is played with a joystick. Plug the joystick into port 1 if you are playing alone (of course, two joysticks are needed for the two-player version). The colored box indicates which square you are on. Use the joystick to position the box on the piece you wish to move, then press the fire button: The box will change color. Now move the box to the square where you want the piece to go, and press the button again. If the move is legal, the piece appears in the new square (if not, you get to try again).

The VIC-20 game requires at least 8K memory expansion and uses keyboard controls exactly like the IBM version. Each square is marked with a letter. When the computer prompts you with FROM and TO, make your move by entering the appropriate letters. Before loading the VIC version, you must enter the following two lines in direct mode (don't add a line number, and hit RETURN after each line):

```
POKE 43,1:POKE 44,32:POKE 8192,0:NEW
POKE 36869,240:POKE 36866,150:POKE
  648,30:PRINT"{CLR}"
```

Atari Version

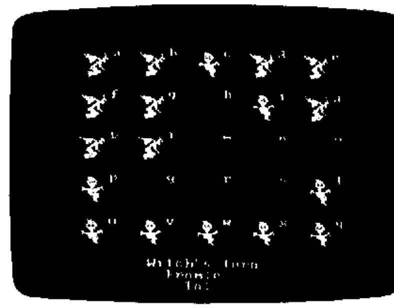
The Atari game requires a joystick (a pair for the two-player game) and is played like the Commodore 64 version. The joystick controls a colored box. Move the box over the piece you want to move, then press the fire button. After the box changes color, move it to the square where you want to put the piece, then press the button again. Player/missile graphics are used to form the witch and ghost figures, and a short machine language routine moves them quickly around the screen.

Apple Version

The Witching Hour runs on any Apple II-series computer with DOS 3.3 or ProDOS. When the program starts, you must choose between a one- or two-player game. Then the game board is drawn and play begins. The flashing box shows which square you are on, and is moved with keyboard controls. Press the I key to go up, J to go left, K for down, and L for right. Press RETURN when the box is on the piece you want to move, then move the box to the desired square and press RETURN again.

TI-99/4A Version

This program runs on any TI-99/4A computer with either console BASIC or TI Extended BASIC. Every game square is labeled with a letter, and the pieces are moved on the board with keyboard controls. The first letter you enter (when the computer prompts FROM:) designates the piece you wish to move. The second letter (entered when the computer prints TO:) designates the square you will move to. The computer signals with a beep when you try an illegal move. The game may be played by one or two players, or the computer can play both sides. Whenever you press ENTER without typing a letter, the computer takes that move.



"The Witching Hour" for IBM PC/PCjr forms ghost and witch shapes with PUT statements.

Program 1: The Witching Hour, PC/PCjr Version

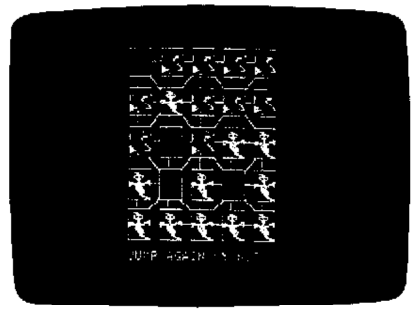
For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" published bimonthly in COMPUTE!.

```
MK 10 GOSUB 530:GOTO 280
OG 20 H=0:K=0:FOR A=7 TO 35:GOS
      B 60:NEXT
GF 30 GOSUB 170:IF H<1 THEN 250
GE 40 H=0:K=0:A=T:GOSUB 60:IF H
      1 THEN 250
HI 50 GOTO 30
DK 60 IF B(A)=0 OR B(A)=-S OR B
      A)=2 THEN RETURN
BP 70 FOR B=0 TO D(A-7):C=A+M(B
      :IF B(C)=S OR B(C)=2 THEN
      160
OC 80 IF B(C) THEN 120
ND 90 SC=RND(1)*.9:IF H<SC THEN
      H=SC:F=A:T=C
AE 100 IF CK=1 AND T1=C THEN L=
       :B=7
EA 110 GOTO 160
NN 120 IF B(C+M(B)) THEN 160
JN 130 SC=1+RND(1)*.9:IF H<SC T
       EN H=SC:F=A:T=C+M(B):K=C
MK 140 IF CK=0 THEN 160
GP 150 IF T1=C+M(B) THEN L=1:K1
       C:B=7
GN 160 NEXT:RETURN
MD 170 B(T)=B(F):B(F)=0:A=F:GOS
       B 760
IF 180 IF K THEN B(K)=0:A=K:GOS
       B 760
DD 190 A=T:GOTO 760
HB 200 GOSUB 520:IF S=1 THEN PR
       NT"The witches win!":GOT
        220
BA 210 PRINT"The ghosts win!"
OF 220 LOCATE 23,10:PRINT"Hit
       key to play again"
OE 230 K$=INKEY$:IF K$="" THEN
       30
GE 240 RUN
NI 250 S=-S:H=0:A=7
LP 260 IF A=36 THEN 200
ML 270 GOSUB 60:IF H=0 THEN A=
       1:GOTO 260
```

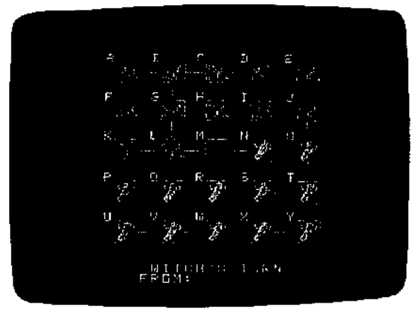
```
E7 650 DIM D(28),B(42),X(35),Y(3
       5),N(28)
0E 660 S = - 1: FOR A = Ø TO 7:
       READ M(A): NEXT : FOR A =
        Ø TO 28: READ D(A): NEXT
53 670 FOR A = Ø TO 4: FOR F = Ø
        TO 4:H = 6 * A + F + 7:X
       (H) = 4 * F + 11:Y(H) = 4
        * A:N(G) = H:G = G + 1:
       NEXT F,A
58 680 FOR A = Ø TO 42: READ B(A
       ): NEXT : FOR A = Ø TO 6:
        READ F: POKE 864 + A,F:
       NEXT : GOSUB 76Ø: GOSUB 1
       19Ø: FOR A = Ø TO 42: GOS
       UB 113Ø: NEXT : RETURN
C8 690 DATA -6,1,6,-1,-5,7,5,-7
44 700 DATA 7,3,7,3,7,Ø,3,7,3,7,
       3,Ø
1A 710 DATA 7,3,7,3,7,Ø,3,7,3,7,
       3,Ø,7,3,7,3,7
A9 720 DATA 2,2,2,2,2,2,2,-1,-1,
       -1,-1,-1,2
44 730 DATA -1,-1,-1,-1,-1,2,-1,
       -1,Ø,1,1,2
BE 740 DATA 1,1,1,1,1,2,1,1,1,1,
       1,2,2,2,2,2,2,2
38 750 DATA 1.Ø.4.Ø.44.62.Ø
A2 760 FOR A = 768 TO 855: READ
       F: POKE A,F: NEXT
7F 770 POKE 6,Ø: POKE 7,141: IF
       PEEK (191 * 256) = 76 THE
       N PRINT CHR$ (4);"PR#A$3Ø
       Ø": GOTO 79Ø
C8 780 POKE 54,Ø: POKE 55,3: CAL
       L 1ØØ2
E3 790 FOR A = 36352 TO 36567: R
       EAD F: POKE A,F: NEXT
18 800 RETURN
2C 1130 IF B(A) = 2 THEN RETURN
22 1140 VTAB Y(A) + 1: HTAB X(A)
45 1150 IF B(A) < Ø THEN PRINT "
        @AB": HTAB X(A): PRINT "
        FGH": HTAB X(A): PRINT "
        LMN"
30 1160 IF B(A) > Ø THEN PRINT "
        CDE": HTAB X(A): PRINT "
        IJK": HTAB X(A): PRINT "
        OPQ"
AB 1170 IF B(A) = Ø THEN PRINT "
        RST": HTAB X(A): PRINT "
        UVW": HTAB X(A): PRINT "
        XYZ"
F3 1180 RETURN
EA 1190 HCOLOR= 3
71 1200 FOR A = 11 TO 139 STEP 3
        2: HPLOT 78,A TO 19Ø,A:
        NEXT
6A 1210 FOR A = 78 TO 19Ø STEP 2
        8: HPLOT A,11 TO A,139:
        NEXT
F8 1220 HPLOT 78,11 TO 194,14Ø:
        HPLOT 194,11 TO 78,14Ø
39 1230 HPLOT 78,76 TO 136,11 TO
         194,76 TO 136,14Ø TO 78
        ,76
2E 1249 RETURN
D2 1250 F = 2:T1 = 2:QS = 2
28 1260 SCALE= QS
83 1270 XDRAW 1 AT (T1 * 4 + 1Ø)
         * 7 - 3,(F * 4) * 8 + 25

3E 1275 PRINT CHR$ (F * 5 + T1 +
         65); CHR$ (8);
49 1280 A$ = "": IF PEEK ( - 163
        84) > 128 THEN GET A$
AF 1285 XDRAW 1 AT (T1 * 4 + 1Ø)
         * 7 - 3,(F * 4) * 8 + 25

F8 1287 QS = QS + 5: IF QS > 27
        THEN QS = 2
EE 1290 IF A$ = "I" AND F > Ø TH
        EN F = F - 1
01 1291 IF A$ = "K" AND F < 4 TH
        EN F = F + 1
BB 1292 IF A$ = "J" AND T1 > Ø T
        HEN T1 = T1 - 1
B7 1293 IF A$ = "L" AND T1 < 4 T
        HEN T1 = T1 + 1
A2 1300 IF A$ < > CHR$ (13) THEN
         126Ø
40 1400 A$ = CHR$ (F * 5 + T1 +
        65): RETURN
D6 1500 DATA 216,12Ø,133,69,134,
        7Ø
2E 1510 DATA 132,71,166,7,1Ø,1Ø
44 1520 DATA 176,4,16,62,48,4
B8 1530 DATA 16,1,232,232,1Ø,134
60 1540 DATA 27,24,1Ø1,6,133,26
A3 1550 DATA 144,2,23Ø,27,165,4Ø
95 1560 DATA 133,8,165,41,41,3
81 1570 DATA 5,23Ø,133,9,162,8
3E 1580 DATA 16Ø,Ø,177,26,36,5Ø
B9 1590 DATA 48,2,73,127,164,36
47 1600 DATA 145,8,23Ø,26,2Ø8,2
9F 1610 DATA 23Ø,27,165,9,24,1Ø5
0F 1620 DATA 4,133,9,2Ø2,2Ø8,226
87 1630 DATA 165,69,166,7Ø,164,7
        1
72 1640 DATA 88,76,24Ø,253
71 1700 DATA 255,129,129,129,129
        ,129
F0 1710 DATA 139,171,255,128,128
        ,192
CB 1720 DATA 192,2Ø8,212,224,255
        ,192
6F 1730 DATA 192,192,194,2Ø2,234
        ,199
9F 1740 DATA 255,129,129,225,129
        ,225
60 1750 DATA 225,225,255,128,128
        ,135
8C 1760 DATA 159,255,238,255,255
        ,192
C9 1770 DATA 192,192,192,192,192
        ,192
80 1780 DATA 171.171.171.169.129
        ,129
A0 1790 DATA 193,199,229,181,181
        ,165
90 1800 DATA 168,17Ø,17Ø,17Ø,199
        ,193
40 1810 DATA 193,193,193,195,199
        ,2Ø4
D1 1820 DATA 129,129,159,255,255
        ,199
B9 1830 DATA 193,193,159,142,142
        ,255
C1 1840 DATA 255,191,191,255,192
        ,248
0C 1850 DATA 255,255,241.192,192
        ,192
55 1860 DATA 223,223,255,159,135
        ,129
72 1870 DATA 129,255,17Ø,17Ø,17Ø
        ,168
0F 1880 DATA 17Ø,139,128,255,216
        ,24Ø
69 1890 DATA 255,193,192,192,192
        ,255
7E 1900 DATA 129,129,129,129,129
        ,129
13 1910 DATA 129,255,255,254,252
        ,248
17 1920 DATA 248,224,128,255,193
        ,193
47 1930 DATA 193,192,193,2Ø7,254
        ,255
39 1940 DATA 127,1,1,1,1,1
46 1950 DATA 1,1,127,Ø,Ø,Ø
D2 1960 DATA Ø,Ø,Ø,Ø,127,64
25 1970 DATA 64,64,64,64,64,64
A4 1980 DATA 1,1,1,1,1,1
53 1990 DATA 1,1,Ø,Ø,Ø,Ø
20 2000 DATA Ø,Ø,Ø,Ø,64,64
FB 2010 DATA 64,64,64,64,64,64
7B 2020 DATA 1,1,1,1,1,1
3B 2030 DATA 1,127,Ø,Ø,Ø,Ø
FF 2040 DATA Ø,Ø,Ø,127,64,64
F6 2050 DATA 64,64,64,64,64,127
```



A ghost is about to be jumped in this game of "The Witching Hour" for Apple II computers.



"The Witching Hour" for the TI-99/4A works with console BASIC as well as TI Extended BASIC.

Program 6: The Witching Hour, TI-99/4A Version

Version by Patrick Parrish, Programming Supervisor

```
100 GOTO 15Ø
110 FOR I=1 TO LEN(H$)
120 CALL HCHAR(R,C+I,ASC(
    SEG$(H$,I,1)))
130 NEXT I
140 RETURN
150 DIM B(42),D(28),N(28)
    ,X(35),Y(35)
160 GOSUB 165Ø
170 GOTO 93Ø
180 H=Ø
190 K=Ø
200 FOR A=7 TO 35
210 GOSUB 31Ø
220 NEXT A
230 GOSUB 6ØØ
240 IF H<1 THEN 86Ø
250 H=Ø
260 K=Ø
270 A=T
280 GOSUB 31Ø
290 IF H<1 THEN 86Ø
300 GOTO 23Ø
310 IF (B(A)<>Ø)*(B(A)<>-
    S)*(B(A)<>2)THEN 33Ø
320 RETURN
330 FOR B5=Ø TO D(A-7)
340 C=A+M(B5)
350 IF (B(C)=S)+(B(C)=2)T
    HEN 58Ø
```

```
360 IF B(C)THEN 450
370 SC=RND*.9
380 IF H>=SC THEN 420
390 H=SC
400 F=A
410 T=C
420 IF (CK<>1)+(T1<>C)THE
    N 580
430 LL=1
440 GOTO 570
450 IF B(C+M(B5))THEN 580
460 SC=1+RND*.9
470 IF H>=SC THEN 520
480 H=SC
490 F=A
500 T=C+M(B5)
510 K=C
520 IF CK=0 THEN 580
530 K=0
540 IF T1<>C+M(B5)THEN 58
    0
550 LL=1
560 K1=C
570 B5=7
580 NEXT B5
590 RETURN
600 A=F
610 B(T)=B(F)
620 B(F)=0
630 GOSUB 2790
640 IF K=0 THEN 680
650 B(K)=0
660 A=K
670 GOSUB 2790
680 A=T
690 GOTO 2790
700 GOSUB 1630
710 IF S<>1 THEN 740
720 H$="THE WITCHES WIN!"
730 GOTO 750
740 H$="THE GHOSTS WIN!"
750 R=23
760 C=9
770 GOSUB 110
780 R=24
790 C=5
800 H$="HIT A KEY TO PLAY
     AGAIN"
810 GOSUB 110
820 CALL KEY(0,KK,SS)
830 IF SS=0 THEN 820
840 GOSUB 2190
850 GOTO 930
860 S=-S
870 H=0
880 A=7
890 IF A=36 THEN 700
900 GOSUB 310
910 A=A+1
920 IF H=0 THEN 890
930 DD=0
940 GOSUB 1630
950 IF S<>1 THEN 980
960 H$="GHOST'S TURN"
970 GOTO 990
980 H$="WITCH'S TURN"
990 R=22
1000 C=10
1010 GOSUB 110
1020 R=23
1030 C=9
1040 H$="FROM:"
1050 GOSUB 110
1060 RANDOMIZE
1070 CALL KEY(0,KK,SS)
1080 IF SS=0 THEN 1060
1090 IF KK<>13 THEN 1120
1100 GOSUB 1630
1110 GOTO 180
1120 IF (KK<65)+(KK>89)TH
     EN 1060
1130 H$=CHR$(KK)
1140 C=15
1150 GOSUB 110
1160 A=N(KK-65)
1170 Z=A
1180 H$="TO:"
1190 IF DD<>1 THEN 1210
1200 CALL HCHAR(23,10,32,
     7)
1210 R=23
1220 C=17
1230 GOSUB 110
1240 CALL KEY(0,KK,SS)
1250 IF SS=0 THEN 1240
1260 H$=CHR$(KK)
1270 C=21
1280 GOSUB 110
1290 T1=N(KK-65)
1300 CK=1
1310 LL=0
1320 K1=0
1330 GOSUB 310
1340 CK=0
1350 H=0
1360 A=7
1370 IF A=36 THEN 1420
1380 GOSUB 310
1390 IF H>=1 THEN 1420
1400 A=A+1
1410 IF A<36 THEN 1380
1420 IF DD THEN 1460
1430 IF LL THEN 1490
1440 CALL SOUND(50,220,5)
1450 GOTO 930
1460 IF (LL<>0)*(K1<>0)TH
     EN 1490
1470 CALL SOUND(50,220,5)
1480 GOTO 1600
1490 IF (K1=0)*(H>=1)THEN
      1440
1500 F=Z
1510 K=K1
1520 T=T1
1530 GOSUB 600
1540 IF K1=0 THEN 860
1550 A=T
1560 Z=A
1570 H=0
1580 GOSUB 310
1590 IF H<1 THEN 860
1600 DD=1
1610 CALL HCHAR(23,22,32)
1620 GOTO 1180
1630 CALL HCHAR(22,1,32,9
     6)
1640 RETURN
1650 FOR I=96 TO 104
1660 READ A$
1670 CALL CHAR(I,A$)
1680 NEXT I
1690 DATA 00000000000000F
     F,0101010101010101,F
     F80808080808080
1700 DATA FF010101010101010
     1,8080808080808080,8
     040201008040201
1710 DATA 010204081020408
     0,01010101010101FF,F
     F00000000000000
1720 FOR I=112 TO 115
1730 READ A$
1740 CALL CHAR(I,A$)
1750 NEXT I
1760 DATA 0000A0FCFE7E3F1
     E,0008080C1C3E1CB8,1
     C090101E1FFE302,F0E8
     CBC4E2FFE000
1770 FOR I=120 TO 123
1780 READ A$
1790 CALL CHAR(I,A$)
1800 NEXT I
1810 DATA 00011131131F030
     7,40F050F4F6F4FCE0,0
     70F0F1F3F3F1C08,E0C0
     800000000000
1820 CALL CLEAR
1830 CALL COLOR(11,4,1)
1840 CALL COLOR(12,15,1)
1850 FOR I=1 TO 8
1860 CALL COLOR(I,16,1)
1870 NEXT I
1880 CALL SCREEN(2)
1890 PRINT TAB(6);"THE WI
     TCHING HOUR"::::::::
     :::::
1900 CALL HCHAR(14,8,112)
1910 CALL HCHAR(14,9,113)
1920 CALL HCHAR(15,8,114)
1930 CALL HCHAR(15,9,115)
1940 CALL HCHAR(14,23,120)
1950 CALL HCHAR(14,24,121)
1960 CALL HCHAR(15,23,122)
1970 CALL HCHAR(15,24,123)
1980 FOR A=0 TO 7
1990 READ M(A)
2000 NEXT A
2010 FOR A=0 TO 28
2020 READ D(A)
2030 NEXT A
2040 FOR A=0 TO 4
2050 FOR F=0 TO 4
2060 H=6*A+F+7
2070 X(H)=4*F+8
2080 Y(H)=4*A+2
2090 N(G)=H
2100 G=G+1
2110 NEXT F
2120 NEXT A
2130 DATA -6,1,6,-1,-5,7,
     5,-7
2140 DATA 7,3,7,3,7,0,3,7
     ,3,7,3,0
2150 DATA 7,3,7,3,7,0,3,7
     ,3,7,3,0,7,3,7,3,7
2160 DATA 2,2,2,2,2,2,2,-
     1,-1,-1,-1,-1,2
2170 DATA -1,-1,-1,-1,-1,
     2,-1,-1,0,1,1,2
2180 DATA 1,1,1,1,1,2,1,1
     ,1,1,1,2,2,2,2,2,2,2,2
2190 CALL COLOR(11,1,1)
2200 S=-1
2210 CALL COLOR(12,1,1)
2220 CALL COLOR(9,1,1)
2230 CALL COLOR(10,1,1)
2240 CALL CLEAR
2250 H$="...SETTING UP GA
     ME BOARD"
2260 R=23
2270 C=3
2280 GOSUB 110
2290 FOR ROW=5 TO 17 STEP
      4
2300 FOR COL=8 TO 24 STEP
      4
2310 CALL HCHAR(ROW.COL.1
     03)
2320 NEXT COL
2330 NEXT ROW
2340 FOR ROW=2 TO 18 STEP
      4
2350 FOR COL=7 TO 23 STEP
      4
2360 CALL HCHAR(ROW,COL+3
     .100)
2370 CALL HCHAR(ROW+1,COL
     +4,99)
2380 CALL HCHAR(ROW,COL,9
     7)
2390 CALL HCHAR(ROW-1,COL
     +2,96)
2400 CALL HCHAR(ROW+1,COL
     +3,98)
2410 CALL HCHAR(ROW+2.COL
     +1,99)
```

```
2420 CALL HCHAR(ROW+2,COL+2,104)
2430 NEXT COL
2440 NEXT ROW
2450 FOR ROW=3 TO 19 STEP 4
2460 CALL HCHAR(ROW,7,97)
2470 CALL HCHAR(ROW,26,100)
2480 CALL HCHAR(ROW,27,32)
2490 NEXT ROW
2500 FOR COL=8 TO 24 STEP 4
2510 CALL HCHAR(1,COL,96,2)
2520 CALL HCHAR(20,COL,104)
2530 NEXT COL
2540 FOR ROW=4 TO 12 STEP 8
2550 FOR COL=10 TO 18 STEP 8
2560 CALL HCHAR(ROW,COL,101)
2570 CALL HCHAR(ROW,COL+5,102)
2580 CALL HCHAR(ROW+1,COL+4,102)
2590 CALL HCHAR(ROW+4,COL+1,102)
2600 CALL HCHAR(ROW+5,COL,102)
2610 CALL HCHAR(ROW+4,COL+4,101)
2620 NEXT COL
2630 NEXT ROW
2640 RESTORE 2160
2650 CALL HCHAR(23,3,32,25)
2660 Q=0
2670 FOR A=0 TO 42
2680 READ B(A)
2690 GOSUB 2790
2700 IF B(A)=2 THEN 2730
2710 CALL HCHAR(Y(A)-1,X(A)-1,Q+65)
2720 Q=Q+1
2730 NEXT A
2740 CALL COLOR(9,14,1)
2750 CALL COLOR(10,14,1)
2760 CALL COLOR(11,4,1)
2770 CALL COLOR(12,15,1)
2780 RETURN
2790 IF B(A)<>2 THEN 2810
2800 RETURN
2810 IF B(A)<>0 THEN 2850
2820 CALL HCHAR(Y(A),X(A),32,2)
2830 CALL HCHAR(Y(A)+1,X(A),32,2)
2840 GOTO 2950
2850 IF B(A)>0 THEN 2910
2860 CALL HCHAR(Y(A),X(A),112)
2870 CALL HCHAR(Y(A),X(A)+1,113)
2880 CALL HCHAR(Y(A)+1,X(A),114)
2890 CALL HCHAR(Y(A)+1,X(A)+1,115)
2900 GOTO 2950
2910 CALL HCHAR(Y(A),X(A),120)
2920 CALL HCHAR(Y(A),X(A)+1,121)
2930 CALL HCHAR(Y(A)+1,X(A),122)
2940 CALL HCHAR(Y(A)+1,X(A)+1,123)
2950 RETURN
```
©

The Beginners Page

Tom R. Halfhill, Editor

Clearing Up Variable Cloudiness

If you're just learning to program, variables can be confusing at first—especially because there are so many varieties of variables. Last month's column introduced the concept of numeric variables. But, depending on your computer's BASIC, there are also integer variables, double-precision variables, string variables, numeric array variables, and string array variables. This month we'll cover integer variables and tackle the rest later.

Numeric variables, you'll recall, represent ordinary numbers. For instance, you can store the number 10 in the variable X with the BASIC statement X = 10. Numeric variables can represent fractions just as easily, as in X = 98.6. An *integer variable* is similar, but with one important difference. As the term implies, integer variables can only represent *integers*—whole numbers. Fractions like 98.6 aren't allowed. There's one other limitation, too. In most BASICs which allow integer variables, the value cannot range beyond a maximum of 32,767 or a minimum of −32,768.

At first, these restrictions may seem odd. What's the advantage of limiting a variable to a whole number, and especially a whole number within a relatively narrow range?

The answer has to do with the way computers manipulate numbers. Internally, they use the binary numbering system instead of our everyday decimal system. Translating decimal numbers into binary gets tricky when the decimal number is a fraction, or *floating point* number (so-called because the decimal point can "float" to the left or right, as in 98.6 or 9.86). The conversion process requires a few valuable microseconds, and it takes several bytes of memory just to store a single floating point number.

Are Integers Faster?

Integer variables can greatly simplify matters for a computer. Because fractions aren't allowed, the operating system doesn't have to spin its wheels performing lengthy floating point conversions. And when the integers are limited to a range of −32,768 to 32,767, each number can be stored in only two bytes of memory.

Saving a few bytes of memory isn't a terribly important consideration anymore, now that nearly all personal computers come with at least 64K of RAM. But on certain computers, integer variables *can* help your programs run faster—often significantly faster.

In Commodore BASIC, Applesoft, and IBM BASIC, you declare an integer variable by appending a percent symbol (%) to the variable name, as in X% = 10. (Integer variables are not available in TI BASIC or Atari BASIC, but are supported in Atari Microsoft BASIC.) A common mistake is to accidentally omit the % symbol in a statement somewhere, often leading to a mysterious error or unexpected result. Keep in mind that two variable names such as X and X% are treated by the computer as completely separate variables—they can store independent values and are as different as A and Z.

To test the performance of integer variables versus regular variables on your computer, enter this simple program:

```
10 FOR X=1 TO 32000
20 Y=Y+1
30 NEXT X
40 PRINT Y
```

Use a watch to measure how long this program takes to execute. Jot down the result, then change all three occurrences of Y to integer variables by adding the % symbol. Now run the program and time it again.

Surprising Results

What happened? If you have an IBM PC or PCjr, the program should run measurably faster. But if you have a Commodore or Apple, the program actually runs *slower*. What's going on?

Integer variables are indeed faster and more memory-efficient on IBM computers. But on Commodore and Apple computers, integer variables actually execute slower and consume just as much memory as regular variables. This is true even though all three computers have versions of Microsoft BASIC. The reason is that the math routines in the Commodore and Apple are designed to handle floating-point numbers only. Therefore, the computer must convert integer variables into floating-point values, perform the math requested by the program, and then convert the results back into integers. All this conversion takes so long (in computer terms) that integer variables really aren't any faster than regular variables on Commodore or Apple computers.

It would seem, then, that integer variables are useless if you have a Commodore or Apple. But in fact, they can speed up your programs and save memory when used to construct *arrays*—a future column topic.

In the meantime, let's clear up another mystery raised by the above program. If you examine it closely, you might wonder why converting Y to Y% makes it run faster even on the IBM. Since the FOR-NEXT loop is incrementing Y by steps of one, Y is never a fraction, anyway—it's always a whole number. But computers handle all numeric variables as floating point numbers, *even when the value is a whole number and not a fraction.* Defining a variable as an integer variable forces the IBM to treat it as an integer. ©

Classified

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.

Advertisers Index