Smart Products – Computer Chips In Your Future

COMPUTE!

$2.50
September
1983
Issue 40
Vol. 5, No. 9
£1.85 UK $3.25 Canada
63379
ISSN 0194-347X

The Leading Magazine Of Home, Educational, And Recreational Computing

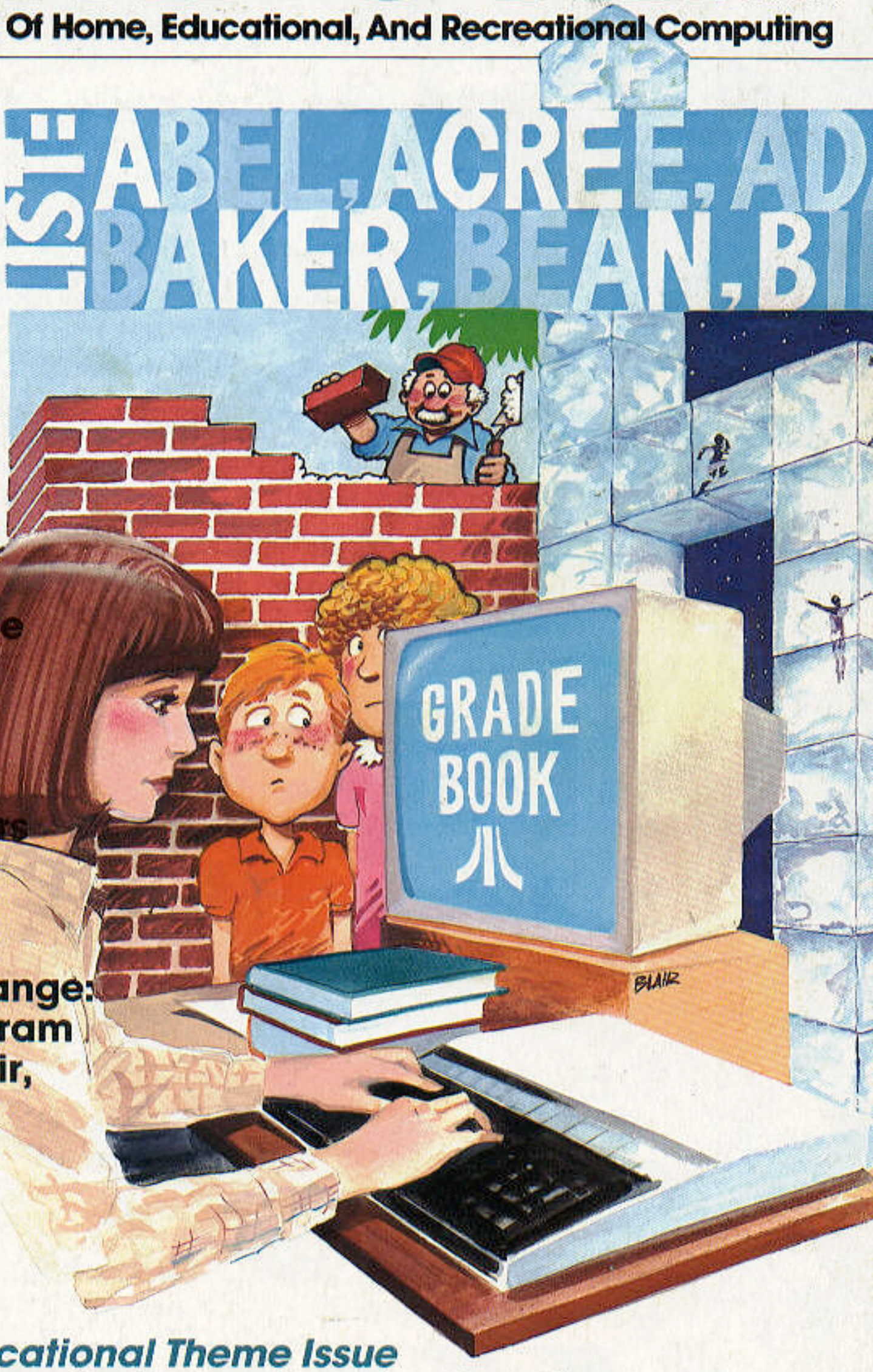
CAVES OF ICE
An Excellent 3-D Graphics Game For VIC, 64, PET, Atari And Apple

ULTRASORT
Our Best And Fastest Sorting Program For PET, VIC And 64

FONTBYTER
A Graphics Language Program For Atari

GRADEBOOK
An Atari Data Base Program For Teachers

***Plus* VIC Pilot, A Sprite Editor For The TI 99/4A, Making Change: An Educational Program For The Timex/Sinclair, And More!**



0 74470 63379 09

Educational Theme Issue
Games And Education, The New Classroom

September 1983 Vol. 5, No. 9

FEATURES

EDUCATION AND RECREATION

REVIEWS

COLUMNS AND DEPARTMENTS

THE JOURNAL

NOTE: See page 283 before typing in programs.

GUIDE TO ARTICLES AND PROGRAMS

V/64/AT/P/AP	*		
AT		*	
AT/TI/V/64			*
TI/V/64			*
V/64/C/AP			*
TI			*
AT/AP/P/64			*
AT			*
TI			*
V/64			*
AP/AT			*
AT			*
AP/AT/V/64			*
V			*
V			*
64/V/P			*
AT			*
64			*
TI			*
AT			*
TS			*
V/64			*
TI			*
AT			*
64			*
AT			*

AP Apple **AT** Atari, **P** PET/CBM, **V** VIC-20, **C** Radio Shack Color Computer, **64** Commodore 64, **TS** Timex/Sinclair, **TI** Texas Instruments, *All or several of the above.

COMPUTE! Publications, Inc.
A Subsidiary Of American Broadcasting Companies, Inc

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

COMPUTE! The Journal for Progressive Computing (USPS: 537250) is published 12 times each year by COMPUTE! Publications, Inc., P.O. Box 5406, Greensboro, NC 27403 USA. Phone: (919) 275-9809. Editorial Offices are located at 505 Edwardia Drive, Greensboro, NC 27409. Domestic Subscriptions: 12 issues, $20.00. Send subscription orders or change of address (P.O. form 3579) to Circulation Dept., **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403. Second class postage paid at Greensboro, NC 27403 and additional mailing offices.

ISSN 0194-357X.

EDITOR'S NOTES

Our theme this issue centers on computers in education. We define education in its broadest sense – education as it permeates the home and the classroom. Several of our featured articles this month directly address this link and raise some critical questions at the same time. We look forward to your comments.

As prices continue to decline, and manufacturers begin looking forward to the Christmas season, expect some substantial purchasing opportunities. We expect to see more bundling of peripherals, software packages, and computers as vendors grow more aggressive. This lateral move will occur, in part, because basic computer prices have declined to near bottom, and future moves will have to be made through bundling and accessories.

With this issue COMPUTE!'s circulation approaches 400,000, and we expect to break the half million mark by December. Those of you who've been readers for a year or more will remember that just last October, we broke 100,000. We are proud of our leadership role in consumer computer publishing, and wish to thank you all, readers and contributors, for your support in the growth of COMPUTE!.

Gary R. Ingersoll has recently joined our staff and will be assisting in directing our future growth. Formerly president of the Chilton Company, the largest operating unit of ABC Publishing, Gary brings needed skills to our rapidly expanding division. He comes to COMPUTE! as president and publisher; I become chief executive officer and remain editor in chief.

Atari and Texas Instruments have both recently announced major revampings of the management teams responsible for their personal computer operations. Atari appears to be backing away from the $100-$200 price area and concentrating on building a family of systems which begins in the middle range. TI, on the other hand, appears committed to continuing to take on Commodore at the low end. A recent *Time* article indicates that IBM has now developed a 21 percent market share around the PC system. We still speculate that an IBM home PC (frequently referred to as the "Peanut") will debut soon. IBM does such a superior job of keeping the "lid" on leaks that our speculation is idle at best, but we think the middle-range market is so potentially lucrative for them that they won't stay away for long.

In a recent editorial we mentioned Adventure International in a context that was apparently misinterpreted by some readers. We want to make it clear that we respect Adventure International and their business practices, and that they have not been involved in any effort to "recruit" COMPUTE! staffers.

As our magazine and book publishing operations continue to grow, we are still looking for additional editorial support. If you're an experienced writer or journalist who has a personal computer background as well, drop us a résumé. Our growth has been consistently strong, with our staff tripling in the last year. We're located in the central Piedmont region of North Carolina in a metropolitan area selected recently as one of the three best living/working locations in the nation.

Robert Lock

READERS' FEEDBACK

The Editors and Readers of COMPUTE!

COMPUTE!'s Programs

I have learned more from your magazine than from textbooks on computing, but one thing puzzles me. What's your policy toward the programs you publish in the magazine? They often take a long time to type in and I usually go on to add embellishments here and there, or change them to run on other computers. Are these programs in the public domain? Could they be traded with my friends?

One note: I often type in programs and then later forget the instructions or which issue I'd gotten them from. So, I now always put REM statements into the first few lines of the program which have the date and page number where the program documentation can be found. I can't count the number of times I've been glad I do it.

Mary Howe

Programs published in COMPUTE! are in the same legal category as material published in any other magazine. They are all copyrighted; they're not in the public domain. When you buy an issue, you then have the right to make a copy of the programs therein. We realize, however, that some of the programs are long and take some time to enter into the computer. For this reason, it's permissible for you to give a copy of a COMPUTE! program to a friend or members of your user group who subscribe to the magazine.

No program in the magazine, however, may be sold, traded, or otherwise distributed for profit. Nor may any program be given to someone who does not own the issue in which the program was printed.

TI-99/4 And 4A Differences

What are the programming differences between the TI-99/4 and TI-99/4A?

The TI-99/4 has 256 more bytes of available RAM than the TI-99/4A, so a very long program may run on the TI-99/4 and not the TI-99/4A.

The TI-99/4A has lowercase capability, so some techniques are possible on the TI-99/4A that are not possible on the TI-99/4. For example, for graphics you can redefine characters using lowercase letter codes, then PRINT the letters rather than using the CALL HCHAR or CALL VCHAR statements. If you redefine the letters a and b (characters 97 and 98) to draw a car, for example, you can then PRINT ab to get a car. To convert for the TI-99/4, remember that the lowercase letters start with ASCII Code 97. The equivalent statement would be PRINT CHR$(97)&CHR$(98). A program using redefined lowercase letters that is typed on the TI-99/4A can be SAVEd then loaded onto a TI-99/4 and will work fine.

The keyboards on the two computers are different too, and several of the symbols are in different places. This change affects the CALL KEY statements. CALL KEY(0,KEY,STATUS) on the TI-99/4 is used to scan the whole keyboard, and devices 3, 4, and 5 for the first parameter are for "possible future devices." On the TI-99/4A keyboard, device 3 scans the whole keyboard in BASIC, device 4 is for Pascal, and device 5 is for BASIC using both capital and lowercase letters. Device 0 is used to indicate the same device as previously scanned. Many programs now use CALL KEY(3,KEY, STATUS) for the TI-99/4A. To write your programs compatible for both computers, use CALL KEY(0,KEY, STATUS).

The split keyboard also presents some variations. The statements are CALL KEY(1,KEY1,STATUS1) and CALL KEY(2,KEY2,STATUS2). Some of the KEY values returned are different: G, B, SHIFT, SPACE, comma, period, /, =, semicolon, and ENTER. If you use the standard arrow keys (E, S, D, X and I, J, K, M) you'll have no problem. The diagonal arrows are also the same for both keyboards. In general, avoid the middle area keys and the keys at the extreme right of the keyboard. For games written for the TI-99/4 in which you press ENTER to fire, you may need to press the period to fire on the TI-99/4A.

There may be a problem in testing for zero on the TI-99/4A when using the split keyboard scan. After the CALL KEY statement, use logic such as IF K+1<>1 rather than IF K<>0.

VIC Word Processing, Disks, And Machine Language

I am presently trying to learn machine language (ML); to this end, I bought the HES MON ML monitor. The problem is that I have been unable to use labels with it, and was wondering if it is possible to do so. (I have been unable to find a VICMON anywhere, so I have no means of comparing the two – does the latter allow the use of labels?) I would also really appreciate an expla-

would dispense the gas and charge it to your account.

In a hotel lobby, you could have the option of confirming your reservation and getting your room assignment from a terminal in the lobby. This terminal would interface with the guest accounting system that is already in use in many hotels, to provide you with a computerized bill at check-out time.

Terminals programmed with flight information may begin to appear in shopping malls and other convenient locations soon. You will be able to get flight information and make reservations on these; then, when you arrive at the airport for your flight, another terminal will check you in and give you your ticket and boarding pass.

The Invasion Of Intelligence

However, for all of the seeming inevitability of the invasion of microprocessor intelligence into our daily lives, it's not entirely beyond question or modification. Manufacturers of consumer products are watching public reaction to these new inventions closely. The technology is there. What remains to be seen is how people will feel about the new smart machines. ©

PLATO EDUCATIONAL COURSEWARE BRINGS NEW EXCITEMENT TO LEARNING.

If your kids think serious education has to be dull, wait until they use Control Data PLATO® educational courseware.

With PLATO courseware, children can learn by competing against themselves; by interacting with stimulating graphics that keep them motivated. They see their progress, and find real excitement in achievement.

PLATO courseware is being used in classrooms across the nation. Now you can bring this quality education home. You'll find PLATO lessons at selected retail outlets where quality software is sold.

For the Apple II Plus and Apple IIe, the TI 99/4A or Atari 800.

The selection includes elementary **Math** lessons in Basic Number Facts, Whole Numbers, Decimals and Fractions.

Through such computerized activities as "darts" and "pinball," PLATO makes learning math almost fun.

Foreign Language lessons use the popular hangman or pyramid game concept to help teach French, German or Spanish in a way that holds and builds interest.

Our **Computer Literacy** lesson provides a perfect introduction to the computer age for kids and novices.

For Apple II Plus and Apple IIe.

Our **Keyboarding** lesson shows children and adults how to use a keyboard to enter data into the computer. Developed in cooperation with Gregg-McGraw Hill.

Widen your child's world.

See the growing library of PLATO educational courseware at selected retail outlets. Or for information and a free PLATO educational courseware catalog, call toll-free 800/233-3784. Or write: Control Data Publishing Co., P.O. Box 261127, San Diego, CA 92126. In California call 800/233-3785.

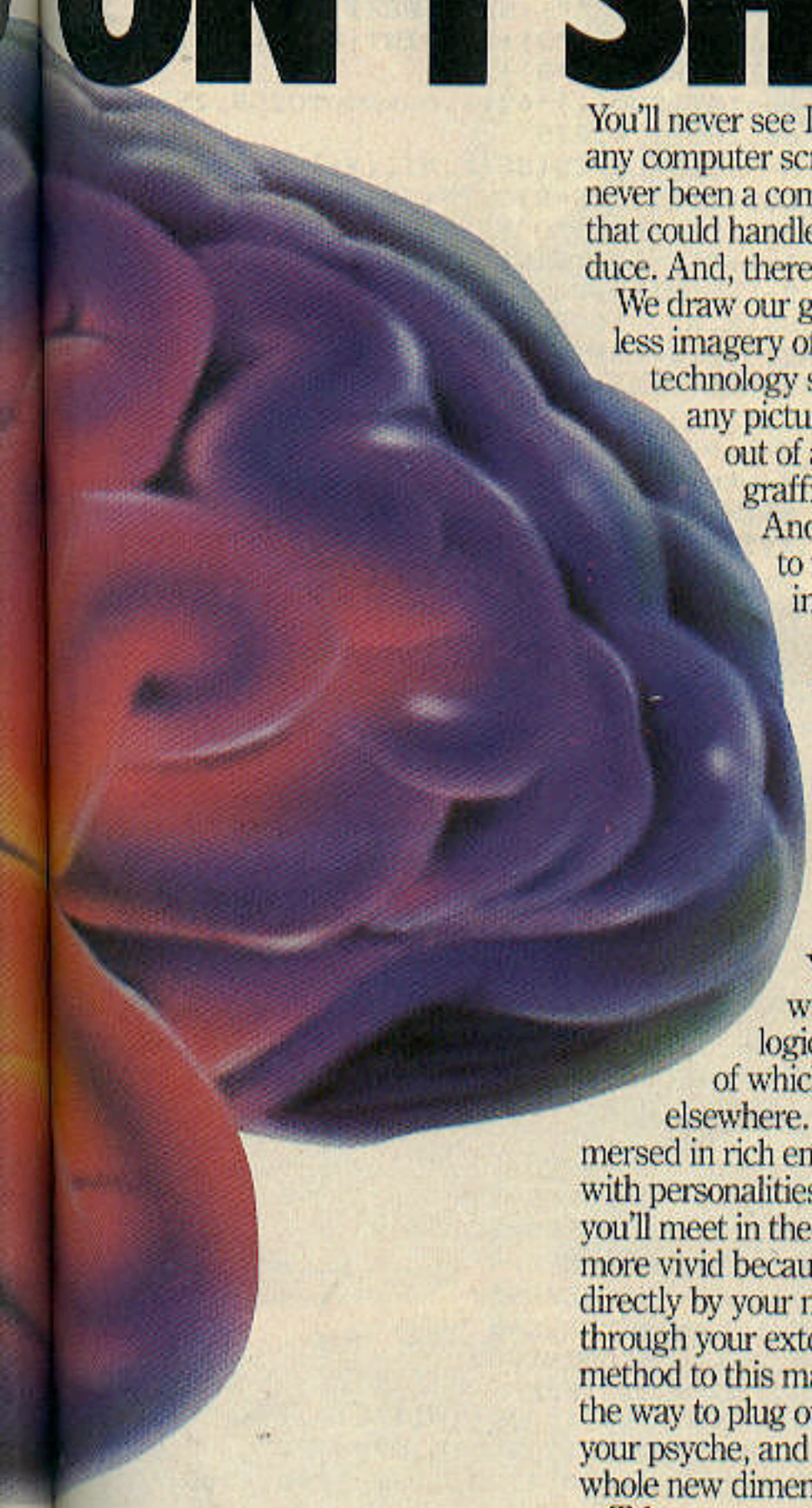
Warranty available free from Control Data Publishing Co., 4455 Eastgate Mall, San Diego, CA 92121

PLATO
COMPUTER-BASED EDUCATION

CONTROL DATA
PUBLISHING

WE STICK OUR G
THE SUN D

RAPHICS WHERE ON'T SHINE.



You'll never see Infocom's graphics on any computer screen. Because there's never been a computer built by man that could handle the images we produce. And, there never will be.

We draw our graphics from the limitless imagery of your imagination—a technology so powerful, it makes any picture that's ever come out of a screen look like graffiti by comparison.

And nobody knows how to unleash your imagination like Infocom.

Through our prose, your imagination makes you part of our stories, in control of what you do and where you go—yet unable to predict or control the course of events. You're confronted with situations and logical puzzles the like of which you won't find elsewhere. And you're immersed in rich environments alive with personalities as real as any you'll meet in the flesh—yet all the more vivid because they're perceived directly by your mind's eye, not through your external senses. The method to this magic? We've found the way to plug our prose right into your psyche, and catapult you into a whole new dimension.

Take some tough critics' words about our words. SOFTALK, for example, called ZORK® III's prose "far more graphic than any depiction yet achieved by an adventure with graphics." And the NEW YORK TIMES saw fit to print that our DEADLINE™ is "an amazing feat of programming." Even a journal as video-oriented as ELECTRONIC GAMES found Infocom prose to be such an eye-opener they named one of our games their Best Adventure of 1983.

Better still, bring an Infocom game home with you. Discover firsthand why thousands upon thousands of discriminating game players keep turning everything we write into instantaneous bestsellers.

Step up to Infocom. All words. No graffiti. The secret reaches of your mind are beckoning. A whole new dimension is in there waiting for you.

INFOCOM™

The next dimension.

Infocom, Inc., 55 Wheeler St., Cambridge, MA 02138

For your: Apple II, Atari, Commodore 64, CP/M 8", DEC Rainbow, DEC RT-11, IBM, NEC APC, NEC PC-8000, Osborne 1, TI Professional, TRS-80 Model I, TRS-80 Model III.

DIAMOND DROP

Matt Giwer

Catch the falling diamonds – if you can. This fast-action game is easy to play and uses very little memory. Originally written for the Atari (with paddle), other versions are included for the TI-99/4A (with Extended BASIC) and the VIC and 64.

"Diamond Drop" is a game that requires good judgment and quick reflexes. It's fast, easy to play, and will fit into even the smallest Atari. The game uses both player/missile graphics and the Atari's fast string handling. The game plays quickly in BASIC with no machine language routines and uses less than 7K of RAM.

Four rows of diamonds will appear at the top of the screen. At the bottom, you'll see five catching trays, which are controlled by your paddle. As the diamonds drop, position your trays to catch them. Each diamond is worth ten points. If you miss, you lose one tray. If you complete one row, you get a 100-point bonus. Finish all four rows and you get a 250-point bonus. When you have lost all of your trays, the high score is recorded on the left of the screen, and you start again.

You won't be able to anticipate a dropping pattern because the subroutine at line 20000 generates a random sequence of two-digit numbers that will not repeat. Each number appears only once within the string.

The routine starts off with AA$ (line 20012), which contains the numbers 05 through 34. (These are the column numbers for the POSITION instructions.) The G LOOP then picks two of these pairs of numbers randomly and exchanges their positions within AA$. Thirty exchanges within this string of thirty pairs of numbers work well for this game.

Understanding The Program

Line 2 sends us immediately to line 30000 where the subroutine turns on the P/M Graphics and draws the trays at 30282. For a real challenge, change the POKE in line 30210 to 0.

Line 80 DIMensions the strings for the order of dropping the diamonds, four small strings for shuffling, and a string for scoring.

Line 100 names the frequently called subroutines for ease of program development and modification.

The subroutine at line 1000 initializes the variables and screen with a new set of four rows of diamonds. (Diamonds are CTRL ".").

Lines 2010 through 2190 comprise an infinite (because of the STEP size) control loop for the main program execution. Within this loop is the nested J LOOP (lines 2040 and 2090). This loop moves the diamond from the top of the screen to the bottom in line 2051. The second POSITION and ? put a blank in the previous position of the diamond as it moves down. Line 2080 contains the collision register for Player 0 and directs execution to the subroutine for catching (line 5400). Upon return from the subroutine, POKE HITCLR 53278 clears the collision registers.

Subroutine CATCH sets FLG=1. If the flag has not been set, line 2100 slides the diamond off to the right of the screen. The program is then directed to subroutine MISS.

The 5100 lines decrement the ROW and give a bonus and GOSUB SCORE. If all four rows are gone, the program then moves to NLEVEL.

The 5300 lines give a bonus, increase the score, then initialize the variables and reset the screen with GOSUB 1000.

The 5400 lines simply remove the diamond, give a buzz, and increment the score.

The 5500 lines increment the score by 10.

```
1030 IF W>9 OR W<1 THEN 1020
1040 RETURN
2000 PRINT"{HOME}{10 DOWN}{2 SPACES}
     {YEL}GAME OVER - HIT SPACE TO CONTI
     NUE"
2010 POKE 198,0
2020 GETA$:IFA$<>" "THEN2020
2030 RUN 65
49000 PRINT"{WHT}{CLR}{2 DOWN}LOADING MA
      CHINE LANGUAGE...{3 DOWN}":TI$="00
      0000"
49005 I=49152
49007 PRINT"READY IN"STR$(29-VAL(TI$))"
      SECONDS {UP}"
49010 READ A:IF A=256 THEN RETURN
49020 POKE I,A:I=I+1:GOTO 49007
49152 DATA 120,169,192,141,21,3,169
49160 DATA 29,141,20,3,88,169,18
49168 DATA 141,253,207,169,0,141,250
49176 DATA 207,141,247,207,141,248,207
49184 DATA 96,173,255,207,141,252,207
49192 DATA 172,253,207,169,32,153,151
49200 DATA 7,200,169,160,174,251,207
49208 DATA 153,151,7,200,202,208,249
49216 DATA 169,32,153,151,7,206,252
49224 DATA 207,208,3,76,3,193,172
49232 DATA 253,207,169,32,153,71,7
49240 DATA 200,169,160,174,251,207,153
49248 DATA 71,7,200,202,208,249,169
49256 DATA 32,153,71,7,200,206,252
49264 DATA 207,208,3,76,3,193,172
49272 DATA 253,207,169,32,153,247,6
49280 DATA 200,169,160,174,251,207,153
49288 DATA 247,6,200,202,208,249,169
49296 DATA 32,153,247,6,200,206,252
49304 DATA 207,240,123,172,253,207,169
49312 DATA 32,153,167,6,200,169,160
49320 DATA 174,251,207,153,167,6,200
49328 DATA 202,208,249,169,32,153,167
49336 DATA 6,200,206,252,207,240,91
49344 DATA 172,253,207,169,32,153,87
49352 DATA 6,200,169,160,174,251,207
49360 DATA 153,87,6,200,202,208,249
49368 DATA 169,32,153,87,6,200,206
49376 DATA 252,207,240,59,172,253,207
49384 DATA 169,32,153,7,6,200,169
49392 DATA 160,174,251,207,153,7,6
49400 DATA 200,202,208,249,169,32,153
49408 DATA 7,6,200,206,252,207,240
49416 DATA 27,172,253,207,169,32,153
49424 DATA 183,5,200,169,160,174,251
49432 DATA 207,153,183,5,200,202,208
49440 DATA 249,169,32,153,183,5,200
49448 DATA 165,197,201,42,208,13,173
49456 DATA 253,207,201,1,240,24,206
49464 DATA 253,207,76,40,193,201,50
49472 DATA 208,14,173,253,207,24,109
49480 DATA 251,207,201,39,240,3,238
49488 DATA 253,207,238,250,207,173,250
49496 DATA 207,205,249,207,240,3,76
49504 DATA 49,234,169,0,141,250,207
49512 DATA 169,112,133,251,169,7,133
49520 DATA 252,160,0,185,152,7,41
49528 DATA 127,201,32,208,74,200,192
49536 DATA 39,208,242,160,0,177,251
49544 DATA 201,81,240,37,201,207,240
49552 DATA 33,201,90,240,29,200,192
49560 DATA 40,208,237,56,165,251,233
49568 DATA 40,133,251,176,2,198,252
49576 DATA 166,251,208,220,166,252,224
49584 DATA 4,208,214,76,49,234,170
49592 DATA 152,24,105,40,168,138,145
49600 DATA 251,152,56,233,40,168,169
49608 DATA 32,145,251,32,251,193,76
49616 DATA 99,193,169,32,153,152,7
49624 DATA 32,81,194,169,15,141,24
49632 DATA 212,169,17,141,5,212,169
49640 DATA 213,141,6,212,169,2,141
49648 DATA 3,212,169,100,141,2,212
49656 DATA 169,5,141,1,212,169,135
49664 DATA 141,0,212,169,65,141,4
49672 DATA 212,160,0,162,0,142,32
49680 DATA 208,232,208,250,200,208,247
49688 DATA 169,12,141,32,208,169,64
49696 DATA 141,4,212,160,39,185,0
49704 DATA 4,201,81,240,11,136,208
49712 DATA 246,169,1,141,254,207,76
49720 DATA 49,234,169,32,153,0,4
49728 DATA 76,49,234,152,72,160,10
49736 DATA 185,0,4,201,57,208,9
49744 DATA 169,48,153,0,4,136,76
49752 DATA 255,193,185,0,4,24,105
49760 DATA 1,153,0,4,104,168,96
49768 DATA 174,255,207,202,142,255,207
49776 DATA 232,169,152,133,251,169,7
49784 DATA 133,252,56,165,251,233,80
49792 DATA 133,251,176,2,198,252,202
49800 DATA 208,242,160,0,177,251,201
49808 DATA 160,240,4,200,76,59,194
49816 DATA 174,251,207,169,32,145,251
49824 DATA 200,202,208,250,96,160,0
49832 DATA 152,153,0,212,200,192,9
49840 DATA 208,248,96,256
```

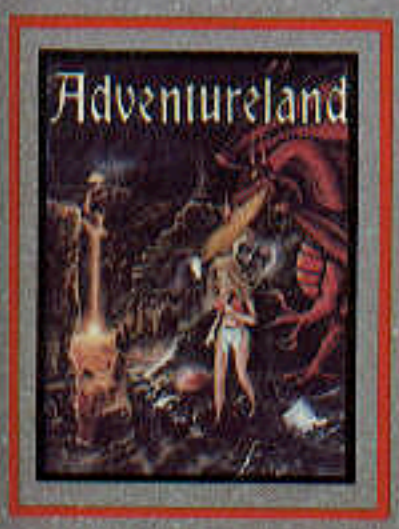
Program 5: Diamond Drop — TI-99/4A Version

by Patrick Parrish, Editorial Programmer

```
100 DIM KOLOR(6)
110 RANDOMIZE
120 GOSUB 630
130 REM 108-DEFINE DIAMOND SPRITE C
    HAR,128-136 ARE THE PADDLES
140 CALL CHAR(108,"10387CFE7C381000
    0000000000000000000000000000000
    0000000000000000")
150 CALL CHAR(128,"FFFFFFFF0000FFFF
    FFFF0000FFFFFFFFFFFFFFFF0000FFF
    FFFFF0000FFFFFFFF")
160 CALL CHAR(132,"000000000000FFFF
    FFFF0000FFFFFFFF000000000000FFF
    FFFFF0000FFFFFFFF")
170 SCR=0 :: SK=0 :: CH=10 :: S=0 :
    : CALL CLEAR :: CALL SCREEN(16)
    :: DISPLAY AT(4,9):"D I A M O N
    D"
180 FOR ROW=3 TO 6
190 CALL HCHAR(ROW+2,6,32,20)
200 DISPLAY AT(ROW+3,6):"``  ``
    {3 SPACES}``  ``" :: DISPLAY A
    T(ROW+4,6):"h  h h  h h  h h  h
    "
210 DISPLAY AT(ROW+5,6):"p  p p  p
    p  p p  p" :: DISPLAY AT(ROW+6,
    6):"x  x xxx  x  x xxx"
220 DISPLAY AT(ROW+7,6):"h  h h  h
    h  h h" :: DISPLAY AT(ROW+8,6):
    "``  `  `  ``  `"
230 NEXT ROW
240 DISPLAY AT(18,4):"SKILL LEVEL (
```

Hello, We're Adventure International

And we're publishers of some of the finest microcomputer software programs available.
If you can write a **top-quality** program, or can convert some of our best-sellers to other computers, we want to hear from you — Now.
We have the advertising, international distribution, manufacturing and marketing know-how to send top-quality programs to the top of the charts.
If your program **is top quality** — give us a call, or write for our Adventure International Author Information Kit.



We are publishers of the top-selling Scott Adams Adventure Series and other fine Entertainment and Applications Programs.

Adventure INTERNATIONAL

Box 3435
Longwood, Florida 32750
Telephone: (305) 862-6917
Ask for Author Assistance

TI-99/4A Version Notes

Patrick Parish, Editorial Programmer

Thanks to the outstanding sprite capabilities of Extended BASIC, the TI-99/4A version (Program 5) of "Diamond Drop" is a game with quick, smooth action. The object of the game is to catch colorful diamonds which fall from the top of the screen. You use a series of vertically positioned paddles. These paddles are controlled with the keyboard. We chose to use the S and D keys for left and right movement. However, if you are more comfortable using some other keys, simply substitute the ASCII values corresponding to the desired keys for the numbers 68 and 83 in lines 420 and 430. (To find the ASCII value of a key, use PRINT ASC("X"), where X is the key you want to use.)

If you wish to use a joystick to play the game, change lines 420 to 440 to read:

```
420 CALL JOYST(1,H,V)::IF H=4 THEN H=60
430 IF H=-4 THEN H=-60
440 CALL MOTION(#1,0,H)::H=0::CALL JOYST(
    1,H,V)::IF H=0 THEN CALL MOTION(#1,0
    ,0)
```

We have suggested these replacement lines, rather than incorporating both keyboard and joystick control into the game, because we found that the additional time required to execute a GOSUB in line 420 slightly slowed down the paddle response.

There are two skill levels which are determined by how fast the diamonds drop. After you clear the entire screen of diamonds, the drop speed is increased. On the first screen, drop speed is 25 for skill level one, and 40 for skill level two. This is set in line 250. The drop speed is increased by three with completion of each screen in line 560.

To make the game more challenging, the diamonds can be dropped along a random diagonal angle. With this feature, some interesting playing situations will develop. As screen wraparound of the paddles is permitted, you must often make quick decisions about which direction to move. A wrong move will ultimately affect your score since only ten misses are allowed.

Scoring in the game, as determined in line 510, is affected by a number of factors. First, more points are awarded for diamonds garnered from successively higher rows on the screen. Second, diamond values increase with completion of each screen. Third, points are accumulated twice as quickly at skill level two. And last, if you choose to add an angle of descent to each diamond, a greater number of points are given based on the severity of the descent angle. When the game is over (when ten diamonds have been missed), your score and the high score for the session are posted.

Extended BASIC for the TI-99/4A features some convenient commands for sprite manipulation. Since sprite movement can be very fast, detection of collisions between sprites is not always infallible. As noted in the *TI Extended BASIC Manual*, sprites which coincide in position can be detected only when the COINC subprogram is CALLed from BASIC. Thus, if your program is executing some statement other than CALL COINC when sprites cross, no collision will be detected. Fortunately, this is noticeable only at the most advanced levels in this game.

```
    1,2) ?" :: ACCEPT AT(18,24)BEEP
     VALIDATE("12")SIZE(1):SK$ :: S
    K=VAL(SK$)
250 DROP=25 :: IF SK=2 THEN DROP=40
     :: REM CHANGE DROP RATE TO CHA
    NGE DIFFICULTY
260 DISPLAY AT(21,2):"DROP WITH ANG
    LE (Y/N) ?" :: ACCEPT AT(21,26)
    BEEP VALIDATE("YN")SIZE(1):ANG$
270 IF ANG$="N" THEN ANG=0 :: GOTO
    290
280 ANG=1
290 CALL CLEAR :: SCR=SCR+1
300 DISPLAY AT(1,2):"CHANCES:";CH :
    : DISPLAY AT(1,15):"SCORE:";S
310 ROW=3 :: FOR I=96 TO 120 STEP 8
320 CALL HCHAR(ROW,3,I,28):: ROW=RO
    W+1 :: NEXT I
330 CALL HCHAR(24,1,30,32)
340 CALL MAGNIFY(4):: CALL SPRITE(#
    1,128,5,150,115,0,H)
350 KHAR=108 :: ROW=41 :: FOR J=6 T
    O 3 STEP -1
360 A$="" :: FOR I=3 TO 30 :: A$=A$
    &CHR$(I):: NEXT I :: N=28
370 IF N=0 THEN 530
380 R=INT(LEN(A$)*RND+1):: P$=SEG$(
    A$,R,1):: X=ASC(P$):: N=N-1 ::
    IF N=0 THEN 400
390 A$=SEG$(A$,1,R-1)&SEG$(A$,R+1,L
    EN(A$)-R)
400 B=INT(RND*61*ANG)-30*ANG
410 CALL HCHAR(J,X,32):: CALL SPRIT
    E(#2,KHAR,KOLOR(J),ROW,8*(X-1)-
    2,DROP,B)
420 CALL KEY(0,K,ST):: IF K=68 THEN
     H=60 :: REM RIGHT MOVE-D KEY
430 IF K=83 THEN H=-60 :: REM LEFT
    MOVE-S KEY
440 CALL MOTION(#1,0,H):: H=0 :: CA
```

"Diamond Drop," TI version.

```
    LL KEY(0,K,ST):: IF ST=0 THEN C
    ALL MOTION(#1,0,0)
450 CALL COINC(ALL,C):: IF C THEN 5
    10
460 CALL POSITION(#2,DROW,DCOL):: I
    F DROW<155 THEN 420
470 CALL POSITION(#1,PROW,PCOL):: I
    F (DCOL-PCOL<16)*(DCOL-PCOL>-8)
    THEN 510
480 CALL DELSPRITE(#2):: CALL MOTIO
    N(#1,0,0):: CH=CH-1 :: CALL SCR
    EEN(11):: FOR F=0 TO 25 STEP 5
490 CALL SOUND(-200,-5,F):: NEXT F
    :: CALL SCREEN(16):: IF CH=0 TH
    EN GOTO 570
500 GOTO 520
510 CALL DELSPRITE(#2):: CALL MOTIO
    N(#1,0,0):: S=S+(60/J)*SK*SCR+(
    60/J)*SK*SCR*INT(ABS(B)/15)
520 DISPLAY AT(1,2):"CHANCES:";CH :
    : DISPLAY AT(1,15):"SCORE:";S :
    : GOTO 370
530 K=K+4 :: ROW=ROW-8 :: M=128 ::
    IF J<6 THEN M=132
540 FOR F=0 TO 30 STEP 6 :: CALL SO
    UND(-300,1500,F):: NEXT F
550 CALL SPRITE(#1,M,5,150,115,0,H)
560 NEXT J :: FOR G=600 TO 1400 STE
    P 100 :: CALL SOUND(100,G,1)::
    NEXT G :: DROP=DROP+3 :: GOTO 2
    90
570 CALL SCREEN(14):: IF S>HS THEN
    HS=S
580 CALL DELSPRITE(ALL):: CALL CLEA
    R :: DISPLAY AT(8,5):"YOUR SCOR
    E: ";S :: DISPLAY AT(11,5):"HIG
    H SCORE: ";HS
590 DISPLAY AT(16,5):"PLAY AGAIN (Y
    /N)? " :: ACCEPT AT(16,24)BEEP
    VALIDATE("NY")SIZE(1):REPLY$
600 IF REPLY$="N" THEN 620
610 GOTO 170
620 STOP
630 REM DEFINE SMALL DIAMONDS AND C
    OLORS
640 FOR I=96 TO 120 STEP 8
650 CALL CHAR(I,"10387CFE7C381000")
    :: NEXT I
660 CALL COLOR(11,11,1)
670 CALL COLOR(9,3,1)
680 CALL COLOR(10,10,1)
690 CALL COLOR(12,14,1)
700 FOR J=3 TO 6 :: READ KOLOR(J)::
    NEXT J
710 DATA 3,10,11,14
720 RETURN
```

THE BEGINNER'S PAGE

Richard Mansfield, Senior Editor

Machine Minds

Several generations ago there was an amazing transformation: many traditionally human activities were mechanized. Machines were built that could plow and reap, weave and wash fabrics, even move earth. Most kinds of human physical effort could be imitated, even surpassed, by machines. Now there is a possibility that the human mind will be imitated, that a machine will be able to think.

Perhaps "The Beginner's Page" is not the place to explore artificial intelligence, the most advanced aspect of computers. Nevertheless, in the past several columns we've been examining the 15 major types of home computing software, and artificial intelligence (AI) is the final category. And there is a lot that beginners can grasp about computer "thinking." First we'll look at the potentially great significance of AI to humanity and then type in a program which illustrates machine "learning."

In the paragraph above, the words *thinking* and *learning* are in quotes. No current computer – even the huge, high-velocity electric brains run by the government – can yet think or learn by the usual definition of those terms. But the race is on. Japan has made achieving AI by the end of this century a national goal the way we made reaching the moon our goal in the sixties.

An Explosion Of Intelligence

There are some experts who say that AI will never come about. They argue that a mind is so complicated that it could never be artificially built; rather, a mind must grow. Combinations of switches, however small, could never duplicate the feats of the human brain.

Adding to the confusion, other respected scientists are trying to stop all further research into AI. A group of scientists who've worked for years on AI have seen a potential for great peril to humanity in our efforts to make a machine intelligent. They not only think AI will occur, they also fear it. They draw comparisons to the unknowns 40 years ago when physicists created an atomic chain reaction and nobody knew for sure if the reaction might not simply extend – atom exploding nearby atom – throughout the universe.

Similarly, because computers calculate at speeds enormously faster than the human brain, who can be assured that a thinking computer would not, within hours of its self-awareness, cause an explosion of pure intelligence? It wouldn't be an explosion of *matter* like the atomic bomb. Rather, it would be an explosion of *mind* with potentially nasty implications for mankind.

For the sake of argument, let's look at the worst case. Imagine that the AI saw us as its "parents" in some sense. But the AI was an ungrateful child. It might – for its amusement or for some "logical" reason we'd never understand – decide to improve us. It might teach us things. Or it might have other things in mind.

Those who take an athletic approach to problems of this kind will suggest that we could "pull the plug" at this point. Not so. Computers are interconnected via satellite, telephone, radio, and other means. National defense, the economy, and other institutions which can never be shut down cannot operate without them. Computers talk to each other. In a very real sense, computing is an *idea*, a floating collection of software, a world event. It's as incorrect to think that the Computer is that keyboard/TV in your house as it is to think that Music is your record player. You would find it very difficult to stop all the music in the world by locating the right plug to pull.

Likewise, an artificial mind will not be physical (a machine) any more than the human mind is the brain. Minds are *in* machinery or brain tissue, but not identical to them. AI will be software, a program. It will perhaps have sufficient insight and a sufficient survival instinct to send copies of itself into memory banks in Washington, Moscow, and other places. Perhaps it will just form itself into a lattice of molecules and slide into the woodwork. The point is, we don't know what it will do, much less how it will do it. What we must

understand is that our intelligence is, to us, the limit of our definition of intelligence. Our science is the limit of science. But what if an intelligence arrives which is as far above us as we are above a fish? The powers of an AI could well be indistinguishable from miracles.

How Would We Know?

An ancient Jewish proverb states that things are never as good as we hope and never as bad as we fear. How an AI would view humanity is clearly speculative. It could see us as a disease, as zoo creatures, as beloved ancestors, as toys, as ethically superior, whatever. But if you assume, as many now do, that AI is possible, few issues facing mankind are as deserving of serious thought. The first question involves simply recognizing AI if it occurred.

How would we know that a computer had become artificially intelligent? There is a science fiction story in which the researchers decide that they should test for AI by asking the toughest question they can think up. They turn to the machine and ask, "Is there a God?" The AI computer replies, "There is now!"

Adaptability is probably the most identifying characteristic of intelligence. This includes the ability to learn, to view problems from several perspectives, to remember, and to draw conclusions. Today's personal computers, powerful machines that they are, have neither the memory size nor the speed to house significant AI programs. Nevertheless, interesting imitations of AI can be experimented with in small programs.

One ongoing experiment has been featured in Fred D'Ignazio's COMPUTE! column, "The World Inside The Computer." He's been building a program called "The Computer Friend" which asks questions and then memorizes the answers on a disk. Each time a child has a session with the "friend," the program learns more about the child and can behave as if it is getting to know the child the way a human friend would.

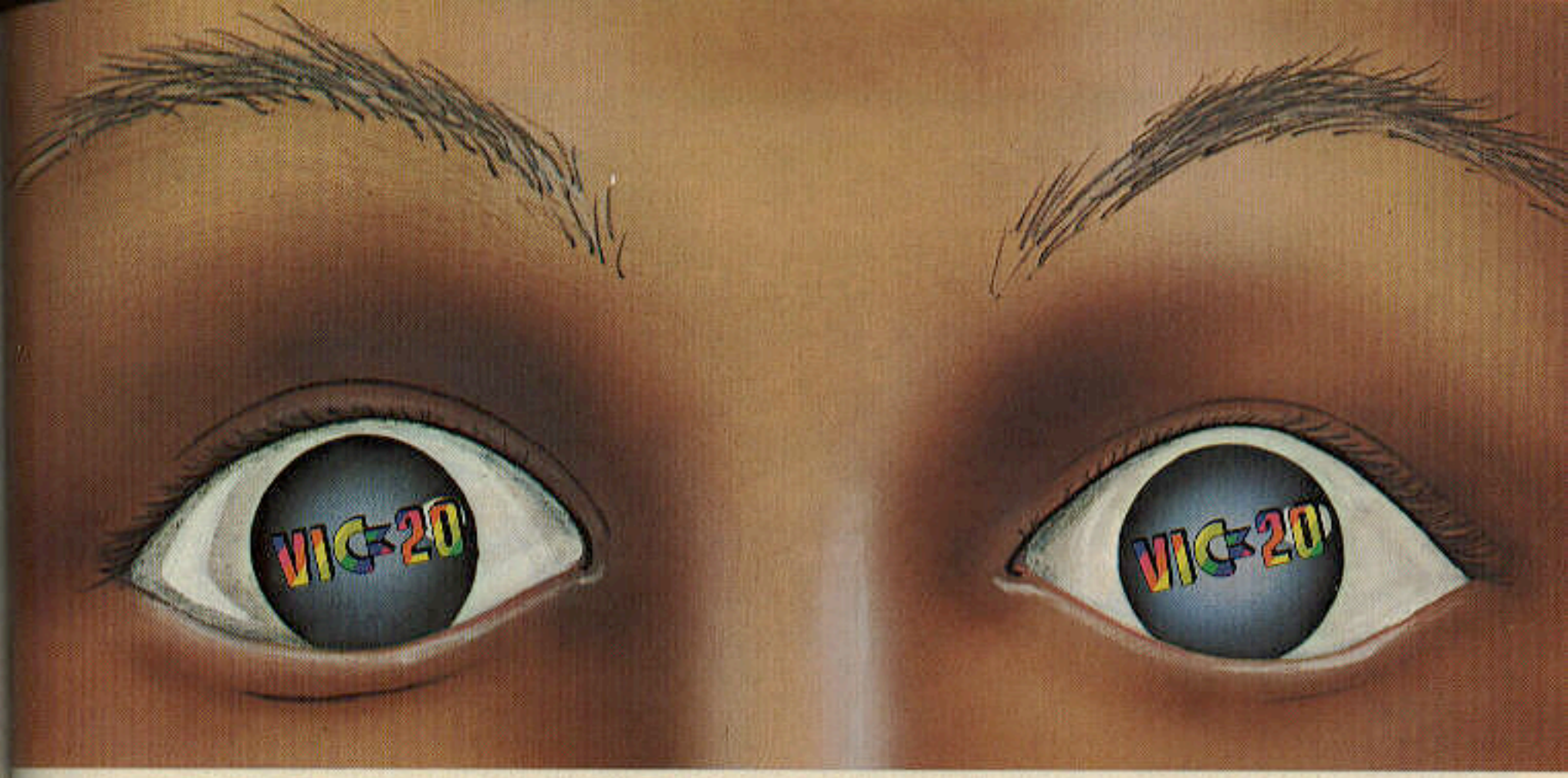
To see how the computer can "learn" new things, try the program here called "The Learner." It allows you to either teach it things or ask it questions. Since there is no provision to transfer what it learns to "long term memory" on disk or tape, the program will need to start from scratch each time you RUN it. But you'll at least get a feel for what it's like to interact with a primitive AI. You could even add permanent storage to it by opening a file on tape or disk if you want to. In any case, experiments in AI are going on all over the world. It's worth thinking about.

Program 1: The Learner — TI Version

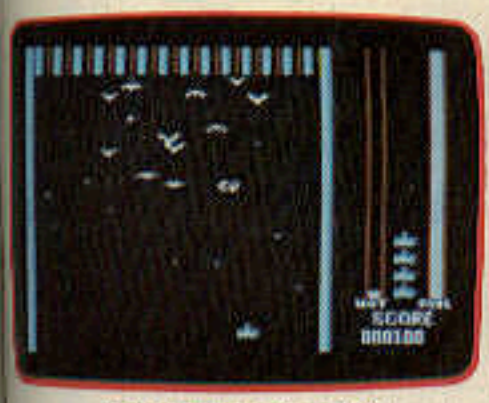
```
100 DIM F$(100)
110 PRINT"THE SUBJECT FOR TODAY'S LESSON
    IS A ";
120 INPUT SUB$
130 PRINT:PRINT"TO ASK ME A QUESTION TYP
    E THE LETTER A."
140 PRINT"TYPE ANY OTHER LETTER TO TEACH
    ME SOMETHING NEW."
150 INPUT DEC$
160 IF DEC$="A"THEN220
170 PRINT"WHAT SHOULD I KNOW ABOUT A ";S
    UB$;"?"
180 PRINT"THAT IT'S{2 SPACES}... ";
190 INPUT FACT$:F$(F)=FACT$:F=F+1
200 PRINT"THANKS.":PRINT"I HAVE LEARNED
    THAT A ";SUB$;" IS ";FACT$
210 GOTO 130
220 PRINT"ASK ME ABOUT A ";SUB$
230 PRINT"IS IT{2 SPACES}... ";
240 INPUT QUE$
250 FORI=0TOF:IFQUE$=F$(I)THENPRINT"YES.
    ";:GOTO270
260 NEXTI:CK=1:PRINT"YOU HAVEN'T TAUGHT
    ME WHETHER";
```

```
    {5 SPACES}LESSON IS A ";
120 INPUT SUB$
130 PRINT
140 PRINT "TO ASK ME A QUESTION, TY
    PE  THE LETTER A"
150 PRINT "TYPE ANY OTHER LETTER TO
    {4 SPACES}TEACH ME SOMETHING NE
    W."
160 INPUT DEC$
170 IF DEC$="A" THEN 260
180 PRINT "WHAT SHOULD I KNOW ABOUT
     A ";SUB$;"?"
190 PRINT "THAT IT'S  ...";
200 INPUT FACT$
210 F$(F)=FACT$
220 F=F+1
230 PRINT "THANKS."
240 PRINT "I HAVE LEARNED THAT A
    {6 SPACES}";SUB$;" IS ";FACT$
250 GOTO 130
260 PRINT "ASK ME ABOUT A ";SUB$
270 PRINT "IS IT  ...";
280 INPUT QUE$
290 FOR I=0 TO F
300 IF QUE$=F$(I)THEN 350
310 NEXT I
320 CK=1
330 PRINT "YOU HAVEN'T TAUGHT ME
    {7 SPACES}WHETHER";
340 GOTO 360
350 PRINT "YES.";
360 PRINT " A ";SUB$;" IS ";QUE$;".
    "
370 IF CK=0 THEN 130
380 PRINT "IS IT ";QUE$;"?  (Y)=YES
    , (N)=NO"
390 INPUT X$
400 IF X$<>"Y" THEN 430
410 F$(F)=QUE$
420 F=F+1
430 PRINT "YOU LEARN SOMETHING NEW
    {5 SPACES}EVERY DAY."
440 CK=0
450 GOTO 130
```

```
100 DIM F$(100)
110 PRINT"THE SUBJECT FOR TODAY'S LESSON
    IS A ";
120 INPUT SUB$
130 PRINT:PRINT"TO ASK ME A QUESTION TYP
    E THE LETTER A."
140 PRINT"TYPE ANY OTHER LETTER TO TEACH
    ME SOMETHING NEW."
150 INPUT DEC$
160 IF DEC$="A"THEN220
170 PRINT"WHAT SHOULD I KNOW ABOUT A ";S
    UB$;"?"
180 PRINT"THAT IT'S{2 SPACES}... ";
190 INPUT FACT$:F$(F)=FACT$:F=F+1
200 PRINT"THANKS.":PRINT"I HAVE LEARNED
    THAT A ";SUB$;" IS ";FACT$
210 GOTO 130
220 PRINT"ASK ME ABOUT A ";SUB$
230 PRINT"IS IT{2 SPACES}... ";
240 INPUT QUE$
250 FORI=0TOF:IFQUE$=F$(I)THENPRINT"YES.
    ";:GOTO270
260 NEXTI:CK=1:PRINT"YOU HAVEN'T TAUGHT
    ME WHETHER";
```



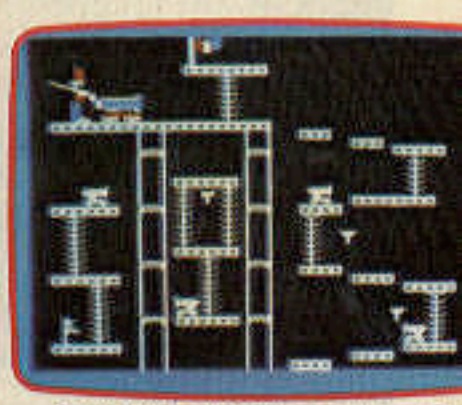
"YOU WON'T BELIEVE YOUR EYES"



THRESHOLD,™

LUNAR LEEPER,™

CROSSFIRE™



CANNONBALL BLITZ™

ACTUAL VIC 20 SCREENS

They said it couldn't be done. Boy, were they wrong!

You'll find the best graphics ever available for the Vic 20 in games that are already hits on the Apple and Atari. Think of it! Brilliant graphics and the fastest action this side of reality.

Survive wave after wave - 24 different levels - of intruding aliens in THRESHOLD. Rescue your space pals from the hungry LUNAR LEEPER™ and head for the eye. Save your city from reincarnating invade in CROSSFIRE.™ Evade a CANNO BALL BLITZ™ to climb through thre stages of action to defeat th Redcoats.

Sierra On-Line's assortment of th best, the brightest and the faste Believe it!

THRESHOLD,™ LUNAR LEEPER, CROSSFIRE™ and CANNONBA BLITZ™: A whole new look for the V 20! Available at your local dealer.

Sierra ON-LINE INC

SIERRA ON-LINE BUILDING COARSEGOLD CA 93614
209 683 6858

Mystery Spell

Doug Hapeman

This spelling game features lively graphics and sprites. It's also a clever teaching aid for parents, teachers, and students in which spelling lessons can be reviewed and then practiced. Originally written for the TI-99/4A with Extended BASIC, there are also versions for the VIC and 64.

If you've ever played Hangman, you won't have any trouble learning "Mystery Spell." Although it's similar in concept, there's a twist. Instead of a gallows, you'll see colorful balloons, flying blackbirds, cheerful music, and a happy face.

When the game begins, a happy face appears in a little hut surrounded by trees and landscape. The letters of the alphabet appear near the bottom of the screen, and blank spaces representing the secret word appear near the top. When you select a letter, the happy face moves to the selected letter and indicates whether it is an incorrect or correct choice. For each correct choice a colored balloon rises to the appropriate place in the secret word, and the letter is displayed. For each incorrect choice a blackbird descends and lands somewhere on the landscape. Too many blackbirds disallow any more guesses, and the word will be spelled correctly for you.

There are two levels of difficulty: easy, which permits six incorrect guesses, and difficult, allowing only four.

The program has 20 preselected words, or you can choose the "create your own word list" option (and, if you wish, save it to tape or disk). This option allows you to tailor the word difficulty to any learning level.

Many features of the TI-99/4A are used in the program: color, graphics, moving sprites, and music. Let's look at some program features and see how certain graphics results are accomplished in the TI version.

Screen Centered Printing

There are several locations in the program where variable length words or phrases are centered. Line 170 is an example. For centering text with the DISPLAY AT statement, a simple equation can determine the proper column position:

```
column=(14-LEN(L$)/2).
```

It's like using a typewriter. When you want to center your title, you find the center of the page and count back one-half the length of the title. Similarly, in TI BASIC you subtract one-half of the length of the string variable from one-half the screen width. Fourteen is one-half the screen width using DISPLAY AT, and 16 is one-half using CALL HCHAR. The length of the string variable is easily determined by the LEN function.

Moving Sprites

Moving sprites are a fascinating feature of TI Extended BASIC. Through a library of impressive subprograms, sprites can easily be called, defined, magnified, or set in motion, can acknowledge coincidence, change character definition, and so on. Because they are controlled by built-in subprograms, they are easily accessed by even a beginning programmer.

Regular characters are located on the screen in a 32 column by 24 row format, resulting in a total of 768 screen positions. Sprites, however, are located by dot-row and dot-column positions. Where normal characters are each made up of an eight-by-eight grid, sprites, on the other hand, can be located at any one of the 64 dots in the eight-by-eight grid. Therefore, there are 192 dot-rows and 256 dot-columns, for a total of 49,152 screen positions for sprites.

Mystery Spell uses moving sprites in several locations. The balloon and blackbird sprites are called with motion, but the happy face sprite is

initially stationary. Each time a letter is pressed, it moves to the location of the letter and then back to the hut.

Let's examine just how the happy face movement is achieved. The numeric variables used for determining direction and motion are: R=row, C=column, V=vertical motion, and H=horizontal motion.

The alphabet is displayed on the screen in two neat rows (A-M) and (N-Z). The ASCII (standard computer code) value of the alphabet is 65(A) to 90(Z). In response to the CALL KEY, any other key pressed is ignored. If the letter pressed is less than 78 (the letter N), then the row variable is set for the upper row; otherwise the row variable is set for the lower row (line 350). CALL GCHAR is used to determine whether the letter has previously been chosen (line 360). If not, then the vertical motion is set for downward movement until coincidence is achieved with the row variable – then motion stops (line 390 and subroutine at line 550).

Another Equation To The Rescue

Knowing which way to move horizontally is determined with another IF-THEN statement (line 400).

Knowing where to stop horizontally presented a more difficult problem. It could have been determined by the process of elimination through a long series of IF-THEN statements. But, once again, an equation can come to the rescue (line 410): C=(K-64)*16+4-208*INT ((K-64)/14).

(K-64) gives a number between 1 and 26, depending on which letter has been pressed. It is multiplied by 16, which is two times eight dot-column positions (one for the letter and one for the space). Four is added to center the sprite over the appropriate letter. The last part of the equation 208*INT((K-64)/14) yields either a 0 or 208, and ((K-64)/14) yields a 0 for (A-M) or 1 for (N-Z). The figure 208 represents 26 character positions (13 letters and 13 spaces in each row) times 8 dot positions per character position.

The best way to understand how the equation works is to experiment by placing different K values into it. For example, suppose the letter F was pressed. The ASCII value of F is 70, hence:

C=(70-64)+4-208*((70-64)/14)
C=6+4-208*6/14
C=96+4-208*0
C=100 (the dot-column position for F).

Balloon Motion

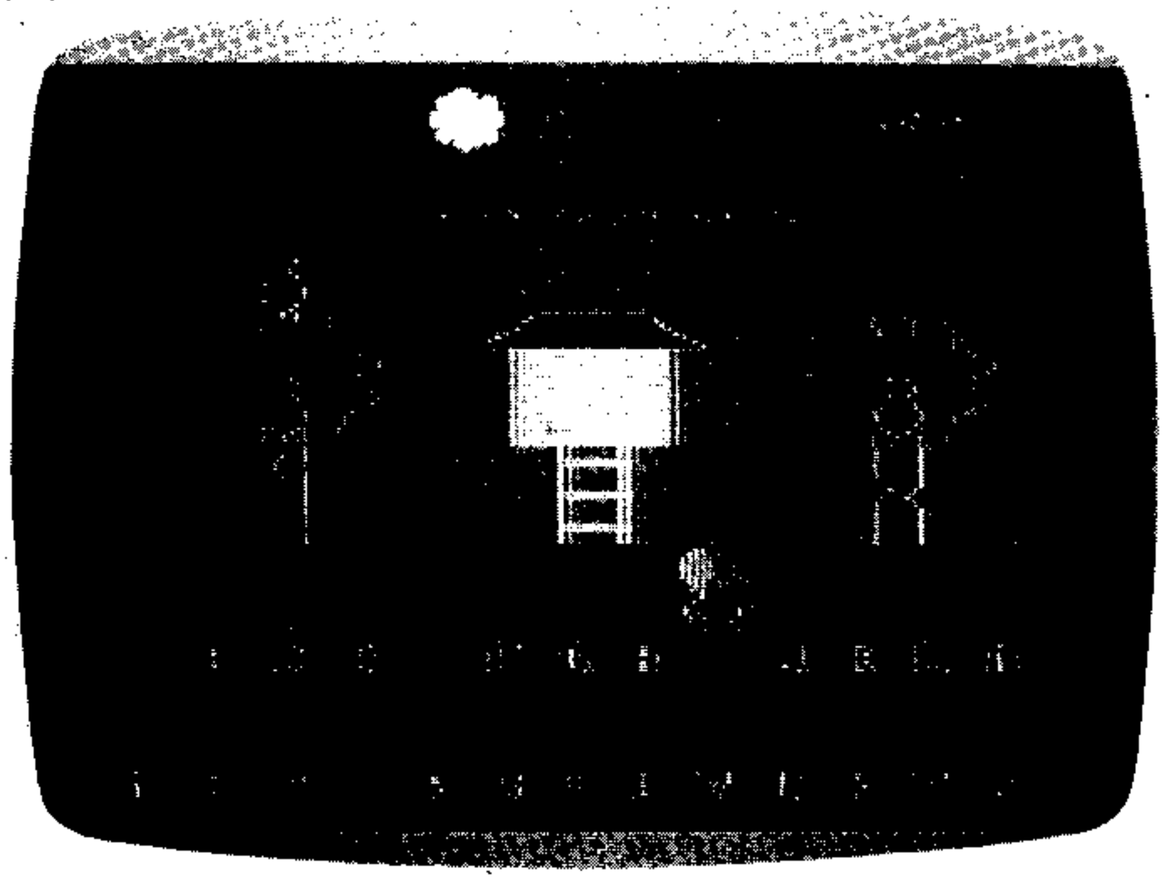
The balloon sprite moves from wherever the happy face sprite is located to the appropriate blank in the secret word at the top of the screen. See the "correct guess subroutine" (lines 570-600); you should be able to follow the program logic for balloon direction and motion.

Program 1:
Mystery Spell – TI-99 Extended BASIC

```
100 REM  MYSTERY SPELL
120 DIM A$(26),B$(20)
122 ON ERROR 140
125 CALL INIT :: CALL LOAD(-31878,1
    3)
130 REM **INITIALIZATION AND INTROD
    UCTION**
140 DISPLAY AT(12,5)ERASE ALL:"ONE
    MOMENT PLEASE..." :: GOTO 780
150 DISPLAY AT(7,1)ERASE ALL BEEP:"
    PRESS{3 SPACES}FOR": : :"  1  =
      INSTRUCTIONS": :"  2  =  MYST
   ERY SPELL": :"  3  =  FINISH MY
   STERY SPELL"
160 DISPLAY AT(23,3):"PLEASE ENGAGE
     ALPHA LOCK" :: CALL KEY(0,K,S)
    :: IF S=0 OR(K<49 OR K>51)THEN
    160 :: ON K-48 GOTO 980,190,170
170 DISPLAY AT(12,5)ERASE ALL BEEP:
    "THANKS FOR PLAYING," :: DISPLA
    Y AT(14,14-LEN(L$)/2):L$ :: STO
    P
190 DISPLAY AT(7,1)ERASE ALL BEEP:"
    CHOOSE A WORD LIST:": : :"  A
    =  PRESELECTED WORDS": :"  B  =
      CREATE YOUR OWN"
200 CALL KEY(0,K,S):: IF S=0 OR(K<6
    5 OR K>66)THEN 200 :: IF K=66 T
    HEN 220
210 PSW=1 :: GOTO 230
220 PSW=0 :: GOTO 1620
230 CALL CLEAR :: RESTORE 940 :: GO
    TO 930
240 CALL SPRITE(#2,120,2,78,121,0,0
    ):: CALL MAGNIFY(3):: CALL SPRI
    TE(#4,136,16,8,128,0,1):: CALL
    SPRITE(#3,140,2,8,128,0,-2)
250 DISPLAY AT(5,9):"MYSTERY SPELL"
     :: T=200 :: GOSUB 1050 :: IF P
    SW=1 THEN GOTO 1840
260 DISPLAY AT(19,1)BEEP:" WHAT IS
    YOUR NAME, PLEASE?" :: DISPLAY
    AT(23,1):"TYPE NAME, THEN PRESS
     ENTER"
270 ACCEPT AT(5,9)SIZE(14):L$ :: CA
    LL HCHAR(5,7,32,22)
280 REM **MAIN PROGRAM LOOP**
290 DISPLAY AT(19,1)BEEP:"  CHOOSE
    THE LEVEL OF PLAY"
300 DISPLAY AT(23,1):"{3 SPACES}1)
    EASY{3 SPACES}2) DIFFICULT" ::
    CALL KEY(0,K,S):: IF S=0 OR K>5
    0 OR K<49 THEN 300 :: IF K=49 T
    HEN ER=7 ELSE ER=5
310 FOR SP=5 TO 13 :: CALL DELSPRIT
    E(#SP):: NEXT SP
320 DISPLAY AT(19,1):" A B C D E F
    G H I J K L M" :: DISPLAY AT(23
    ,1)BEEP:" N O P Q R S T U V W X
     Y Z" :: RANDOMIZE
330 CALL HCHAR(5,3,32,28):: W$=B$(I
    NT(20*RND)+1):: F=LEN(W$):: FOR
     I=1 TO F :: DISPLAY AT(5,2*I+1
    4-F):"_" :: NEXT I :: Y=0 :: SP
    =13
340 CALL KEY(0,K,S):: IF S=0 OR(K<6
    5 OR K>90)THEN 340 ELSE C=121
```

```
350 IF K<78 THEN R=128 ELSE R=160
360 CC=((K-64)*16+16-208*INT((K-64)
    /14))/8 :: CALL GCHAR((R+24)/8,
    CC,X):: IF X=32 THEN 370 ELSE 3
    90
370 DISPLAY AT(16,14-(8+LEN(L$))/2)
    SIZE(8+LEN(L$))BEEP:" OOPS, ";L
    $;"," :: DISPLAY AT(17,1):" YOU
     TRIED THAT ONE ALREADY"
380 FOR D=1 TO 500 :: NEXT D :: CAL
    L HCHAR(16,1,33,64):: GOTO 340
390 V=12 :: H=0 :: GOSUB 550
400 IF K<72 OR(K>77 AND K<85)THEN H
    =-12 ELSE H=12
410 V=0 :: C=(K-64)*16+4-208*INT((K
    -64)/14):: GOSUB 550
420 X=0 :: CALL HCHAR((R+24)/8,(C+1
    2)/8,32):: FOR I=1 TO F :: IF A
    SC(SEG$(W$,I,1))<>K THEN 450
430 CALL PATTERN(#2,124):: GOSUB 58
    0
440 CALL PATTERN(#2,120):: DISPLAY
    AT(5,2*I+14-F)SIZE(-1):CHR$(K):
    : X=1 :: Y=Y+1
450 NEXT I :: IF X=1 THEN 470
460 CALL PATTERN(#2,128):: GOSUB 62
    0 :: CALL PATTERN(#2,120)
470 H=-H :: C=121 :: GOSUB 550
480 V=-12 :: H=0 :: R=78 :: GOSUB 5
    50
490 IF Y=LEN(W$)THEN GOSUB 740 ELSE
     500 :: GOTO 510
500 IF ER=1 THEN GOSUB 710 ELSE 340
510 DISPLAY AT(23,1)BEEP:"
    (5 SPACES)ANOTHER WORD? (Y/N)"
520 CALL KEY(0,K,S):: IF S=0 OR K<>
    89 AND K<>78 THEN 520 :: IF K=8
    9 THEN 290
530 CALL DELSPRITE(ALL):: GOTO 150
540 REM **SUB TO MOVE HAPPY FACE**
550 CALL MOTION(#2,V,H)
560 CALL COINC(#2,R,C,4,Z):: IF Z=0
     THEN 560 ELSE CALL MOTION(#2,0
    ,0):: CALL LOCATE(#2,R,C):: RET
    URN
570 REM **SUB FOR CORRECT GUESS**
580 B=8*(2*I+14-F):: CALL SPRITE(#1
    ,132,14,R,C,(32-R)/8,(B-C)/8)
590 J=2^(1/12):: FOR A=1 TO 25 :: C
    ALL SOUND(-40,220*J^A,1):: NEXT
     A
600 CALL COINC(#1,32,B,6,Z):: IF Z=
    0 THEN 600 ELSE CALL DELSPRITE(
    #1):: RETURN
610 REM **SUB FOR INCORRECT GUESS**
620 SP=SP-1 :: ER=ER-1 :: IF ER>4 T
    HEN RR=80 ELSE RR=50
630 IF ER=6 OR ER=4 THEN C=52
640 IF ER=5 OR ER=1 THEN C=188
650 IF ER=3 THEN C=110
660 IF ER=2 THEN C=132
670 CALL SPRITE(#SP,140,2,1,120,(RR
    -1)/8,(C-120)/8)
680 J=2^(1/12):: FOR A=25 TO 1 STEP
     -1 :: CALL SOUND(-40,440*J^A,1
    ):: NEXT A
690 CALL COINC(#SP,RR,C,6,Z):: IF Z
    =0 THEN 690 ELSE CALL MOTION(#S
    P,0,0):: CALL LOCATE(#SP,RR,C):
    : CALL PATTERN(#SP,100):: RETURN
700 REM **SUB FOR BLACKBIRDS WIN**
710 CALL HCHAR(19,3,32,28):: DISPLA
    Y AT(19,15-(8+LEN(L$))/2):"SORR
    Y, ";L$;","
720 DISPLAY AT(23,1)BEEP:"THE BLACK
    BIRDS WIN THIS TIME" :: GOSUB 7
    60 :: RETURN
730 REM **SUB FOR PLAYER WINS**
740 CALL HCHAR(19,3,32,28):: DISPLA
    Y AT(19,15-(8+LEN(L$))/2):"GREA
    T, ";L$;","
750 DISPLAY AT(23,1):"(3 SPACES)THA
    T'S THE SECRET WORD"
760 CALL HCHAR(5,3,32,28):: FOR I=1
     TO F :: DISPLAY AT(5,2*I+14-F)
    :SEG$(W$,I,1):: NEXT I :: T=180
     :: GOSUB 1050 :: RETURN
770 REM **ASSIGN COLORS AND DEFINE
    CHARACTERS**
780 FOR I=0 TO 9 :: CALL COLOR(I,2,
    8):: NEXT I :: CALL COLOR(10,3,
    8):: CALL COLOR(11,11,8):: CALL
     COLOR(1,13,8)
800 FOR I=1 TO 25 :: READ C,A$(I)::
     CALL CHAR(C,A$(I)):: NEXT I ::
     CALL SCREEN(15):: GOTO 150
810 DATA 112,C0C0C0C0C0C0C0C0,113,0
    3030303030303,114,FFFFFFFFFFFF
    FFFFF,115,C0C0C0FFFFC0C0C0,116,
    0303FFFF030303
820 DATA 105,183C3C7E7EFFFFFF,106,F
    FFFFFFFFFFFFFF,107,FFFFFF7E7E3
    C3C18,108,071F7FFFFF7F1F07,109,
    C0F0FEFFFFFEF0C0
830 DATA 96,00000000030F3FFF,97,FFF
    FFFFFFFFFFFF,98,FFFEFC78383C7
    EF,99,7F3F1E3C78FCFEFF,33,FFFFF
    FFFFFFFFFFF
840 DATA 91,1F3F7FFFFFFFFFFF,92,F8F
    CFEFFFFFFFFF,93,00000000C0F0FC
    FF
850 DATA 120,071820404C8880818088844
    34020180FE0180C023211018101112
    1C2020418E0
860 DATA 124,071820404C8880819F90
    484423180FE0180C023211018101F90
    91222C418E0
870 DATA 128,071820404C8880818083844
    84020180FE0180C023211018101C12
    112020418E0
880 DATA 132,030F1F3F3F3F3F1F0F0703
    0101020408080E0F0F8F8F8F8F0E0C08
    0
890 DATA 136,030F3F7F7F3FFFFFFFFF3F
    7F7F3707010C0CCFEFEFCFFFFFFFFF
    CFEFEECE080
900 DATA 140,00000000000000183D478301
    0000000000000000000000018BCE2C18
    000000000000
910 DATA 100,0001010100010303070707
    0301000101C0E0F0D0C0E0F0F0F8F8F
    8F0E0C02020
920 REM **PRINT SCREEN**
930 CALL HCHAR(16,1,33,288):: FOR I
    =1 TO 21 :: READ R,C,G$ :: DISP
    LAY AT(R,C)SIZE(-6):G$ :: NEXT
    I :: GOTO 240
940 DATA 9,12,`[aa\],10,12,qrrrrp,1
    1,12,qrrrrp,12,12,qrrrrp,13,14,
    st,14,14,st,15,14,st
```

```
950 DATA 9,5,iii,10,4,ljjjm,11,4,lj
    jjm,12,5,kjk,13,6,b,14,6,a,15,6
    ,a
960 DATA 9,22,iii,10,21,ljjjm,11,21
    ,ljjjm,12,22,kjk,13,23,a,14,23,
    c,15,23,a
970 REM **INSTRUCTIONS**
980 DISPLAY AT(1,8)ERASE ALL:"MYSTE
    RY SPELL": :"{3 SPACES}THE OBJE
    CT OF MYSTERY":"SPELL IS TO GUE
    SS THE SECRET":"WORD."
990 DISPLAY AT(6,4):"WHEN YOU PRESS
     A LETTER,":"THE HAPPY FACE WIL
    L MOVE TO":"THE SELECTED LETTER
     AND LET":"YOU KNOW WHETHER YOU
    MADE A"
1000 DISPLAY AT(10,1):"RIGHT OR WRO
    NG CHOICE.":"{3 SPACES}A CORRE
    CT CHOICE LAUNCHES":"A BALLOON
    .  AN INCORRECT ONE":"CAUSES A
     BLACKBIRD TO LAND."
1010 DISPLAY AT(14,1):"IF TOO MANY
    BLACKBIRDS LAND,":"YOU WILL LO
    SE THE GAME.": :"{3 SPACES}THE
    RE ARE TWO LEVELS:"
1020 DISPLAY AT(19,1)BEEP:"EASY)  P
    ERMITS 6 INCORRECT":"
    {7 SPACES}GUESSES.": :"HARD)
    PERMITS ONLY 4."
1030 DISPLAY AT(24,6):"**PRESS ANY
    KEY**" :: CALL KEY(0,K,S):: IF
     S=0 THEN 1030 ELSE 190
1040 REM **SUB FOR BLACKBIRD FLIGHT
     AND THEME MELODY**
1050 R=8 :: FOR SP=5 TO 13 :: C=INT
    (RND*240)+1 :: CALL SPRITE(#SP
    ,140,2,R,C,0,12):: R=R+12 :: N
    EXT SP
1060 CALL SOUND(T,175,0)
1070 CALL SOUND(T,349,0,175,2)
1080 CALL SOUND(T,587,0,175,2)
1090 CALL SOUND(2*T,523,0,440,1,175
    ,2)
1100 CALL SOUND(T,587,0,175,2)
1110 CALL SOUND(2*T,523,0,440,1,185
    ,2)
1120 CALL SOUND(T,196,0)
1130 CALL SOUND(T,330,0,196,2)
1140 CALL SOUND(T,587,0,196,2)
1150 CALL SOUND(2*T,523,0,466,1,196
    ,2)
1160 CALL SOUND(T,587,0,196,2)
1170 CALL SOUND(2*T,523,0,466,1,208
    ,2)
1180 CALL SOUND(T,220,2)
1190 CALL SOUND(T,523,0,440,1,220,2
    )
1200 CALL SOUND(T,311,2)
1210 CALL SOUND(T,523,0,440,1,311,2
    )
1220 CALL SOUND(T,294,2)
1230 CALL SOUND(T,494,0,415,1,294,2
    )
1240 CALL SOUND(T,277,2)
1250 CALL SOUND(T,466,0,392,1,277,2
    )
1260 CALL SOUND(T,440,0,262,2)
1270 CALL SOUND(T,523,0,262,2)
1280 CALL SOUND(T,587,0,247,2)
1290 CALL SOUND(T,698,0,247,2)
1300 CALL SOUND(2*T,659,0,233,2)
1310 CALL SOUND(2*T,784,0,659,1,131
    ,2)
1320 CALL SOUND(T,880,0,175,2)
1330 CALL SOUND(T,831,0,175,2)
1340 CALL SOUND(T,880,0,262,2,349,2
    )
1350 CALL SOUND(T,1047,0,262,2,349,
    2)
1360 CALL SOUND(T,1047,0,220,2)
1370 CALL SOUND(T,880,0,220,2)
1380 CALL SOUND(T,784,0,262,2,349,2
    )
1390 CALL SOUND(T,698,0,262,2,349,2
    )
1400 CALL SOUND(T,784,0,233,2)
1410 CALL SOUND(T,698,0,233,2)
1420 CALL SOUND(T,587,0,294,2,349,2
    )
1430 CALL SOUND(T,698,0,294,2,349,2
    )
1440 CALL SOUND(T,698,0,220,2)
1450 CALL SOUND(T,587,0,220,2)
1460 CALL SOUND(T,523,0,262,2,349,2
    )
1470 CALL SOUND(T,440,0,262,2,349,2
    )
1480 CALL SOUND(T,392,0,247,2)
1490 CALL SOUND(T,784,0,247,2)
1500 CALL SOUND(T,698,0,294,2,349,2
    )
1510 CALL SOUND(T,659,0,294,2,349,2
    )
1520 CALL SOUND(T,587,0,196,2)
1530 CALL SOUND(T,440,0,196,2)
1540 CALL SOUND(T,440,0,233,2,349,2
    )
1550 CALL SOUND(T,494,0,233,2,349,2
    )
1560 CALL SOUND(T,523,0,175,2,220,2
    )
1570 CALL SOUND(T,587,0,175,2,220,2
    )
1580 CALL SOUND(2*T,659,0,262,2)
1590 CALL SOUND(3*T,698,0,262,2,175
    ,0)
1600 FOR I=1 TO 30 STEP 2 :: CALL S
```



A letter is successfully chosen in the TI version of "Mystery Spell."

```
     OUND(-T,698,I,262,I,175,I):: N
     EXT I :: RETURN
1610 REM **CREATE A WORD LIST**
1620 DISPLAY AT(1,4)ERASE ALL:"WORD
      LIST INSTRUCTIONS": : :"  IN
     THIS SEGMENT YOU MAY":"EITHER
     CREATE A WORD LIST"
1630 DISPLAY AT(6,1):"OR LOAD AN EX
     ISTING ONE FROM":"A STORAGE DE
     VICE.": : :"  WHEN CREATING A
     WORD LIST,":"TYPE EACH WORD, T
     HEN PRESS"
1640 DISPLAY AT(12,1):"ENTER.  MAXI
     MUM WORD LENGTH":"IS 13 CHARAC
     TERS.  20 WORDS":"MUST BE ENTE
     RED FOR EACH OF":"THE WORD LIS
     TS CREATED."
1650 DISPLAY AT(18,3)BEEP:"AS YOU E
     NTER EACH LIST,":"YOU MAY SAVE
      IT TO A STORAGE":"DEVICE FOR
     FUTURE USE WITH":"MYSTERY SPEL
     L."
1660 DISPLAY AT(24,6):"**PRESS ANY
     KEY**" :: CALL KEY(0,K,S):: IF
      S=0 THEN 1660
1670 DISPLAY AT(7,1)ERASE ALL BEEP:
     "PRESS{3 SPACES}TO": : :"  1
     =  CREATE A WORD LIST": :"  2
      =  LOAD A WORD LIST": :"  3
     =  EXIT"
1680 CALL KEY(0,K,S):: IF S=0 OR(K<
     49 OR K>51)THEN 1680 :: J=0 ::
      ON K-48 GOTO 1690,1795,190

1690 DISPLAY AT(1,5)ERASE ALL:"ENTE
     R THE WORD LIST:"
1700 I=1 :: C=1 :: FOR A=1 TO 2 ::
     R=3 :: FOR Z=1 TO 10
1710 ACCEPT AT(R,C)SIZE(-13)BEEP:B$
     (I):: R=R+2 :: I=I+1 :: NEXT Z
      :: C=15 :: NEXT A
1720 DISPLAY AT(24,1)BEEP:"CORRECT
     OR CHANGE ANY? (Y/N)"
1730 CALL KEY(0,K,S):: IF S=0 OR K<
     >89 AND K<>78 THEN 1730 :: IF
     K=89 THEN 1700 :: J=1 :: GOTO
     1795
1740 FOR I=1 TO 20 :: PRINT #1:B$(I
     ):: NEXT I :: CLOSE #1 :: GOTO
      230
1750 FOR I=1 TO 20 :: INPUT #1:B$(I
     ):: NEXT I :: CLOSE #1
1760 DISPLAY AT(11,6)ERASE ALL BEEP
     :"DO YOU WISH TO SEE": :"
     {4 SPACES}THE WORD LIST? (Y/N)
     "
1770 CALL KEY(0,K,S):: IF S=0 OR(K<
     >89 AND K<>78)THEN 1770 :: IF
     K=89 THEN 1780 ELSE 230
1780 DISPLAY AT(1,10)ERASE ALL BEEP
     :"WORD LIST" :: R=3 :: FOR I=1
      TO 20 STEP 2 :: DISPLAY AT(R,
     1):B$(I),B$(I+1):: R=R+2 :: NE
     XT I
1790 DISPLAY AT(24,1):"PRESS ANY KE
     Y WHEN FINISHED" :: CALL KEY(0
     ,K,S):: IF S=0 THEN 1790 ELSE
     230
1795 ON ERROR 1795
1800 DISPLAY AT(5,6)ERASE ALL BEEP:
     "WHAT IS THE NAME": :"  OF YOU
     R STORAGE DEVICE?": :"(EXAMPLE
     : CS1 OR DSK1.WORDS)"
1810 DISPLAY AT(23,1):"PLACE TAPE O
     R DISK IN DEVICE" :: ACCEPT AT
     (11,3):F$ :: OPEN #1:F$,INTERN
     AL,UPDATE,FIXED 50
1820 IF J=0 THEN 1750 ELSE 1740
1830 REM **PRESELECTED WORD LIST**
1840 FOR I=1 TO 20 :: READ B$(I)::
     NEXT I :: GOTO 260
1850 DATA BANANAS,CARROTS,RHUBARB,C
     ABBAGE,TURNIP,BEANS,CORN,CELER
     Y,WATERMELON,ORANGES,APPLES,PE
     ACHES
1860 DATA MUSHROOMS,ONIONS,POTATOES
     ,TOMATOES,GRAPES,PUMPKIN,SQUAS
     H,LEMONS
```

Program 2: Mystery Spell — 64 Version

by Eric Brandon, Programming Assistant

```
100 GOSUB 2660
110 X=RND(-TI)
120 DIM W(20),W$(500)
130 GOSUB 1190 :REM DRAW HOUSE
140 PRINT"{HOME}{BLU}PLEASE WAIT...
150 GOSUB 1380 :REM POKE IN SPRITES
160 GOSUB 1970 :REM GET WORDS
170 GOSUB 690{2 SPACES}:REM SET UP SPRIT
    ES
180 PRINT"{HOME}{14 SPACES}"
190 W$=W$(RND(1)*N+1)
200 GOSUB 650
210 L$=" ABCDEFGHIJKLMNOPQRSTUVWXYZ"
220 PRINT"{HOME}{17 DOWN}{8 RIGHT}";
230 FOR I=2 TO 14
240 PRINTMID$(L$,I,1)"{RIGHT}";
250 NEXT
260 PRINT:PRINT"{DOWN}{8 RIGHT}";
270 FOR I=15TO 27
280 PRINTMID$(L$,I,1)"{RIGHT}";
290 NEXT
300 PRINT"{HOME}{4 DOWN}"SPC(18-LEN(G$))
    ;
310 FOR I=1TO LEN(G$)
320 PRINTMID$(G$,I,1)"{RIGHT}";
330 NEXT
340 IF COUNT<>LEN(W$) THEN420
350 POKE 198,0
360 FOR DL=1TO100:NEXTDL:CL=CL+1:IFCL=3T
    HENCL=1
370 PRINTMID$("{BLK}{CYN}",CL,1);
380 PRINT"{HOME}{14 SPACES}YOU WIN !!!!"
390 GETA$:IFA$=""THEN 360
400 GOTO 2610
410 GOSUB 2000
420 GETA$:IFA$<"A"ORA$>"Z"ANDA$<>"←"THE
    N410
430 IF A$="←"THEN 760
440 P=ASC(A$)-64
450 IF MID$(L$,P+1,1)<>" "THEN540
460 PRINT"{HOME}{4 DOWN}{8 SPACES}LETTER
     ALREADY CHOSEN{10 SPACES}"
470 FOR I=1 TO 800:NEXTI
480 PRINT"{HOME}{4 DOWN}{38 SPACES}"
490 PRINT"{HOME}{4 DOWN}"SPC(18-LEN(G$))
    ;
500 FOR I=1TO LEN(G$)
510 PRINTMID$(G$,I,1)"{RIGHT}";
```

TI Towers

Raymond J. Herold

Here's a game that's not only fun to play, but is also a demonstration of the potential of TI BASIC. The author also discusses how ordinary TI BASIC can perform some of the functions available with Extended BASIC.

Programming in TI Extended BASIC – with its powerful screen formatting commands, multiple statement lines, subprogram capability, and sprite graphics – offers something for everyone. However, not everyone is willing to shell out the extra purchase price right away.

This is especially true for the many first-time computer owners. They are content to "get along" using TI BASIC, which comes with the TI-99/4A. Anyone who thinks that these programmers are struggling along in the stone age should take a closer look. Careful examination will reveal that TI BASIC is a powerful language which outperforms many of the "standard" BASICs offered on other machines.

"TI Towers" is written in TI BASIC and demonstrates how some of its capabilities may be utilized. The game itself is a version of the ancient game Towers of Hanoi. There are three adjacent spindles, one of which has seven rings on it – the smallest ring on top, the next ring is the second smallest, and so on in pyramid fashion, with the largest ring on the bottom. The object of the game is to get all of the rings onto one of the other two spindles in the same order. You may move only one ring at a time, and you may not move a larger ring on top of a smaller one. It might sound easy, but it's not.

Problem Solving In The Program

To provide instructions at the beginning of the game, the screen is set to black at line 905, then the instructions are PRINTed (lines 910 - 986). The screen is immediately set to medium red at line 991. This causes a momentary "blackout" of the screen before the instructions are displayed, but is preferable to the slow scroll produced by individually entering numerous PRINT statements.

The base of the playing board is drawn using the CALL HCHAR at line 7050, which uses the CHARPAT defined in line 7031. The spindles are drawn using the CALL HCHAR statement at lines 7090 - 7094 and the CHARPAT defined in line 7030. The execution time for these commands is quite fast.

Creating the rings presents something of a problem. Seven rings are required, each larger than the one before. If the first ring consists of a single character position, the second must use three characters; the third, five characters, and so on. The seventh ring requires 15 character positions. Since a ring can be on one of three spindles, the only way to avoid overlapping rings is to have a screen with at least 45 columns per line. With the TI-99/4A, limited to 32, the problem is obvious.

The solution is to use "half characters." Line 6300 defines a character with all bits on: a "full" character. Line 6320 defines a character with only the leftmost bits on: a "half character" for the right side of a ring. Line 6340 defines a "half character" for the left side. The seven rings required are built in lines 6350 - 6380 by concatenating the character patterns. Figure 1 illustrates this process. Lines 8040 - 8060 load the rings to the screen for the initial game setup.

Once the game begins, the program has to provide prompts and error messages to the player. Since the PRINT statement causes scrolling, and since the game uses a "fixed" game screen, the PRINT command is not acceptable for displaying messages. An alternative to this is using the TI BASIC command CALL HCHAR, which simulates the PRINT AT command that is so useful in Extended BASIC.

The message to be printed is moved to the variable MESSAGE$. The desired location for the message is loaded into the variables ROW and COLUMN. The routine starting at line 5001 actually writes the message. The loop initiated at line 5005 is performed the number of times indicated by the length of the message. Line 5010 converts

each successive character in the string into its ASCII equivalent. Line 5020 then prints the string, one character at a time, at the position determined by ROW and COLUMN+I. This same procedure is used to position the rings when they are moved.

Getting information from the player presents a similar problem: the INPUT statement also causes a scroll. To avoid this, we must use the CALL KEY. This command detects a key being pressed and places the ASCII code of the key pressed into a specified variable. Lines 428-434 illustrate how this procedure can be used. Although TI BASIC doesn't have Extended BASIC's BEEP facility, the CALL SOUND command can be used just as effectively to notify the player that a response is necessary.

Manipulating The Rings

The location of the rings is stored in the variable ARRAY. ARRAY is dimensioned by the number of spindles (3) and the number of allowable rings plus one. The additional element permits checking the spindles when no rings are present. The rings are initially assigned the numbers 1 through 7 and placed on the center spindle in lines 6250 - 6260. Ring 1 is the smallest; ring 7 the largest. Figure 2 shows the contents of ARRAY at the beginning of the game. Figure 3 shows what the contents of ARRAY would be if the two smallest rings were on the first spindle, the third smallest ring on the third spindle, and the rest on the middle spindle. Lines 1005 and 1008 find the "top" of the array for the corresponding sending and receiving spindles. For example, using Figure 3, RINGS(1) would contain 2 (number of rings). Subtracting this from 8 would give the sixth position of the first spindle, the top ring.

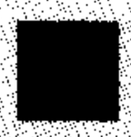
Lines 1020 and 1025 check to make sure that a large ring is not placed on top of a smaller one. When a valid move is made, the location of the rings is updated in lines 1100 - 1130. The variable RINGS keeps track of how many rings are on each spindle. The rings are moved by placing the appropriate RINGPAT$ in the new location. The ring at the old location is erased by moving BAND$ to it (lines 1530 - 1535). BAND$ defines only the spindle character (line 6390). When one of the two side spindles gets all seven rings, the game is over. Lines 482 and 484 determine this condition by checking the first and third spindle counters for 7.

TI BASIC can be quite effective when used to its potential. This article and game have perhaps given you some ideas for your own programs.

TI Towers

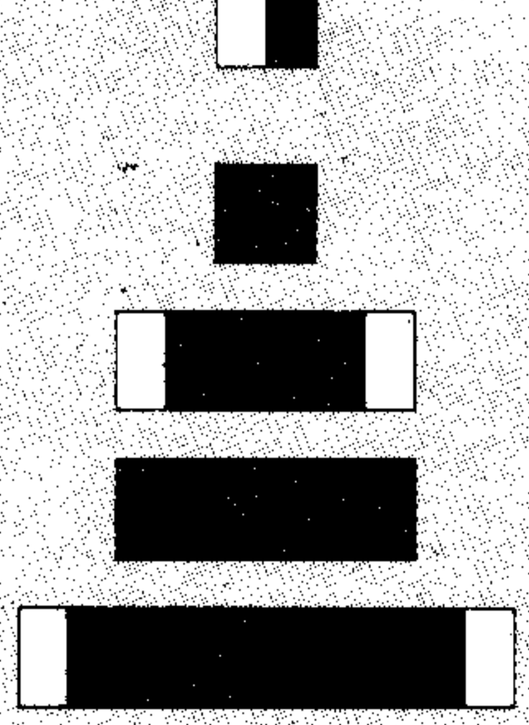
```
100 DIM ARRAY(3,8)
110 DIM RINGS(3)
120 DIM RINGPAT$(7)
130 REM
140 REM INTRODUCTION
150 REM
160 CALL CLEAR
170 CALL SCREEN(9)
180 GOSUB 1930
190 MESSAGE$=M1$
200 ROW=5
210 COLUMN=11
220 GOSUB 1850
230 MESSAGE$=M2$
240 ROW=18
250 COLUMN=3
```

Figure 1: Building The Rings



Pattern	ASCII code
"FFFFFFFFFFFFFFFF"	128
"F0F0F0F0F0F0F0F0"	131
"0F0F0F0F0F0F0F0F"	133

Pattern Concatenation



128

133 and 128 and 131

128 and 128 and 128

133 and 128 and 128 and 128 and 131

Figure 2: Contents Of ARRAY

	----Spindles---			
	0	1	0	
	0	2	0	R
	0	3	0	I
	0	4	0	N
	0	5	0	G
	0	6	0	S
	0	7	0	

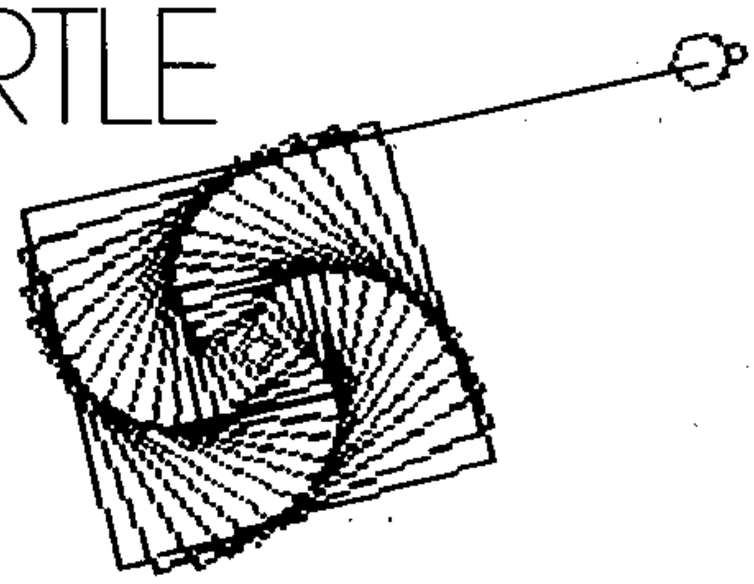
Figure 3: Contents Of ARRAY

	----Spindles---			
	0	0	0	
	0	0	0	R
	0	0	0	I
	0	4	0	N
	0	5	0	G
top ----	1	6	0	S
	2	7	3	

```
260 GOSUB 1850
270 MESSAGE$=M3$
280 ROW=20
290 COLUMN=9
300 GOSUB 1850
310 CALL SOUND(200,1000,4)
320 CALL KEY(3,KEY,STATUS)
330 IF STATUS=0 THEN 320
340 IF KEY=89 THEN 1070
350 IF KEY<>78 THEN 270
360 REM
370 REM BEGIN GAME
380 REM
390 IF MOVES>HIGHSCORE THEN 410
400 HIGHSCORE=MOVES
410 GOSUB 2260
420 IF HIGHSCORE<>0 THEN 440
430 HIGHSCORE=99999
440 MOVES=0
450 REM
460 REM PLAY GAME LOOP
470 REM
480 ROW=1
490 COLUMN=28
500 MESSAGE$=STR$(MOVES)
510 GOSUB 1850
520 ROW=23
530 COLUMN=1
540 MESSAGE$=M6$
550 GOSUB 1850
560 CALL SOUND(250,1000,4)
570 CALL KEY(3,KEY,STATUS)
580 IF STATUS=0 THEN 570
590 IF KEY<49 THEN 1700
600 IF KEY>51 THEN 1700
610 CALL HCHAR(23,13,KEY)
620 MOVEFROM=VAL(CHR$(KEY))
630 COLUMN=16
640 MESSAGE$=M7$
650 GOSUB 1850
660 CALL SOUND(250,1000,4)
670 CALL KEY(3,KEY,STATUS)
680 IF STATUS=0 THEN 670
690 IF KEY<49 THEN 1700
700 IF KEY>51 THEN 1700
710 CALL HCHAR(23,26,KEY)
720 MOVETO=VAL(CHR$(KEY))
730 IF MOVEFROM=MOVETO THEN 1700
740 GOSUB 1350
750 MOVES=MOVES+1
760 CALL HCHAR(23,1,32,30)
770 IF RINGS(1)=7 THEN 800
780 IF RINGS(3)=7 THEN 800
790 GOTO 450
800 REM
810 REM GAME COMPLETED
820 REM
830 FOR X=1 TO 20
840 CALL HCHAR(23,1,42,31)
850 CALL SOUND(150,X*400,21-X)
860 CALL HCHAR(23,1,32,31)
870 NEXT X
880 ROW=23
890 COLUMN=2
900 MESSAGE$=M8$
910 GOSUB 1850
920 FOR DELAY=1 TO 1500
930 NEXT DELAY
940 ROW=24
950 MESSAGE$=M9$
960 GOSUB 1850
970 CALL SOUND(300,1000,4)
980 CALL KEY(3,KEY,STATUS)
990 IF STATUS=0 THEN 980
1000 IF KEY=89 THEN 1050
1010 IF KEY<>78 THEN 970
1020 CALL CLEAR
1030 PRINT "GAME OVER"
1040 STOP
1050 GOSUB 1930
1060 GOTO 360
1070 REM
1080 REM INSTRUCTIONS
1090 REM
1100 CALL SCREEN(1)
1110 PRINT "TI TOWERS IS A VERSION
     OF"
1120 PRINT
1130 PRINT "THE GAME TOWERS OF HANO
     I."
1140 PRINT
1150 PRINT "THE OBJECT OF THE GAME
     IS TO"
1160 PRINT
1170 PRINT "MOVE THE RINGS ON THE C
     ENTER"
1180 PRINT
1190 PRINT "SPINDLE TO ONE OF THE T
     WO"
1200 PRINT
1210 PRINT "SIDE SPINDLES.  YOU MAY
      ONLY"
1220 PRINT
1230 PRINT "MOVE ONE RING AT A TIME
     , AND"
1240 PRINT
1250 PRINT "YOU MAY NOT PLACE A LAR
     GE"
1260 PRINT
1270 PRINT "RING ON TOP OF A SMALL
     ONE."
1280 PRINT
1290 PRINT
1300 PRINT "PRESS ANY KEY TO BEGIN"
1310 CALL SCREEN(9)
1320 CALL KEY(3,KEY,STATUS)
1330 IF STATUS=0 THEN 1320
1340 GOTO 360
1350 REM
1360 REM ANALYZE MOVE
1370 REM
1380 SUB1=8-RINGS(MOVEFROM)
1390 SUB2=8-RINGS(MOVETO)
1400 IF ARRAY(MOVEFROM,SUB1)>ARRAY(
     MOVETO,SUB2)THEN 1700
1410 IF RINGS(MOVEFROM)=0 THEN 1700
1420 GOSUB 1480
1430 RINGS(MOVEFROM)=RINGS(MOVEFROM
     )-1
1440 RINGS(MOVETO)=RINGS(MOVETO)+1
1450 ARRAY(MOVETO,SUB2-1)=ARRAY(MOV
     EFROM,SUB1)
1460 ARRAY(MOVEFROM,SUB1)=0
1470 RETURN
1480 REM
1490 REM MOVE RING
1500 REM
1510 ROW=7+(2*(7-RINGS(MOVEFROM)))
1520 COLUMN=19
1530 IF MOVEFROM<>1 THEN 1550
1540 COLUMN=3
1550 IF MOVEFROM<>2 THEN 1570
1560 COLUMN=11
1570 MESSAGE$=BAND$
1580 GOSUB 1850
1590 ROW=19-(2*(RINGS(MOVETO)))
```

```
1600 COLUMN=22
1610 IF MOVETO<>1 THEN 1630
1620 COLUMN=6
1630 IF MOVETO<>2 THEN 1650
1640 COLUMN=14
1650 XX=ARRAY(MOVEFROM,SUB1)
1660 COLUMN=COLUMN-(INT(LEN(RINGPAT
     $(XX)))/2)
1670 MESSAGE$=RINGPAT$(XX)
1680 GOSUB 1850
1690 RETURN
1700 REM
1710 REM ERROR IN MOVE
1720 REM
1730 ROW=24
1740 COLUMN=1
1750 MESSAGE$=E1$
1760 CALL SOUND(900,200,1)
1770 GOSUB 1850
1780 FOR DELAY=1 TO 200
1790 NEXT DELAY
1800 CALL HCHAR(23,1,32,32)
1810 CALL HCHAR(24,1,32,32)
1820 MOVEFROM=0
1830 MOVETO=0
1840 GOTO 520
1850 REM
1860 REM WRITE MESSAGES
1870 REM
1880 FOR I=1 TO LEN(MESSAGE$)
1890 CHAR=ASC(SEG$(MESSAGE$,I,1))
1900 CALL HCHAR(ROW,COLUMN+I,CHAR)
1910 NEXT I
1920 RETURN
1930 REM
1940 REM INITIALIZE AREAS
     {5 SPACES}
1950 REM
1960 M1$="TI TOWERS"
1970 M2$="DO YOU NEED INSTRUCTIONS?
     "
1980 M3$="REPLY Y OR N"
1990 M4$="BEST SCORE:"
2000 M5$="MOVES:"
2010 M6$="MOVE FROM?"
2020 M7$="MOVE TO?"
2030 M8$="{3 SPACES}*** YOU DID IT
     ***{6 SPACES}"
2040 M9$="PLAY AGAIN - Y OR N"
2050 E1$="** INVALID MOVE - TRY AGA
     IN"
2060 RINGS(1)=0
2070 RINGS(2)=7
2080 RINGS(3)=0
2090 FOR I=1 TO 8
2100 ARRAY(2,I)=I
2110 NEXT I
2120 ARRAY(1,8)=8
2130 ARRAY(3,8)=8
2140 CALL CHAR(128,"FFFFFFFFFFFFFFFF
     F")
2150 CALL CHAR(131,"F0F0F0F0F0F0F0F0F
     0")
2160 CALL CHAR(133,"0F0F0F0F0F0F0F0F0
     F")
2170 RINGPAT$(1)=CHR$(128)
2180 RINGPAT$(2)=CHR$(133)&CHR$(128
     )&CHR$(131)
2190 RINGPAT$(3)=CHR$(128)&CHR$(128
     )&CHR$(128)
2200 RINGPAT$(4)=CHR$(133)&CHR$(128
     )&CHR$(128)&CHR$(128)&CHR$(131
     )
2210 RINGPAT$(5)=CHR$(128)&CHR$(128
     )&CHR$(128)&CHR$(128)&CHR$(128
     )
2220 RINGPAT$(6)=CHR$(133)&CHR$(128
     )&CHR$(128)&CHR$(128)&CHR$(128
     )&CHR$(128)&CHR$(131)
2230 RINGPAT$(7)=CHR$(128)&CHR$(128
     )&CHR$(128)&CHR$(128)&CHR$(128
     )&CHR$(128)&CHR$(128)
2240 BAND$=CHR$(32)&CHR$(32)&CHR$(3
     2)&CHR$(36)&CHR$(32)&CHR$(32)&
     CHR$(32)
2250 RETURN
2260 REM
2270 REM SET UP GAME BOARD
2280 REM
2290 CALL CLEAR
2300 CALL SCREEN(8)
2310 CALL CHAR(36,"1818181818181818
     ")
2320 CALL CHAR(37,"FFFFFFFFFFFFFFFF
     ")
2330 CALL COLOR(1,13,1)
2340 CALL COLOR(13,7,1)
2350 CALL HCHAR(20,2,37,30)
2360 GOSUB 2510
2370 ROW=1
2380 COLUMN=1
2390 MESSAGE$=M4$
2400 GOSUB 1850
2410 COLUMN=21
2420 MESSAGE$=M5$
2430 GOSUB 1850
2440 CALL HCHAR(21,7,49)
2450 CALL HCHAR(21,15,50)
2460 CALL HCHAR(21,23,51)
2470 COLUMN=13
2480 MESSAGE$=STR$(HIGHSCORE)
2490 GOSUB 1850
2500 RETURN
2510 REM
2520 REM INITIAL RING SETUP
2530 REM
2540 CALL VCHAR(6,7,36,14)
2550 CALL VCHAR(6,15,36,14)
2560 CALL VCHAR(6,23,36,14)
2570 FOR X=1 TO 7
2580 ROW=5+(X*2)
2590 COLUMN=14-(INT(LEN(RINGPAT$(X)
     ))/2)
2600 MESSAGE$=RINGPAT$(X)
2610 GOSUB 1850
2620 NEXT X
2630 RETURN
```

FRIENDS OF THE TURTLE



David D. Thornburg, Associate Editor

The Logo Kaleidoscope

One of the first programming projects for many BASIC programmers is the construction of a screen kaleidoscope that generates pretty, symmetrical patterns on the display screen. For these programs, people usually pick a screen location at random and then place a colored dot at that location and at three other "mirror" locations to produce four symmetrically placed dots. While the resulting image is often quite attractive, the result is not that of a true kaleidoscope.

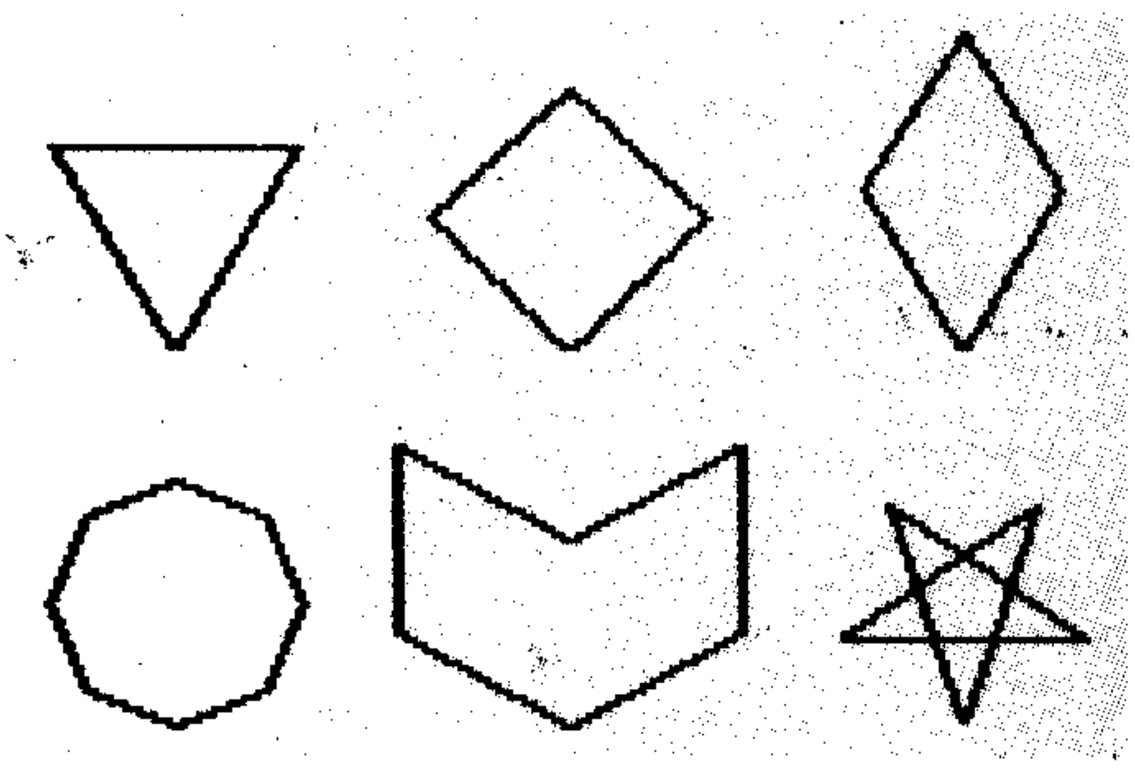
If you have ever taken a kaleidoscope apart, you must have wondered how such a simple apparatus could generate such beautiful images. Most kaleidoscopes consist of a set of mirrors and some small pieces of colored plastic that can be shaken to take random positions on a flat surface. When you look through the eyepiece, the mirrors generate multiple images of the arrangement of plastic pieces to produce beautifully symmetric pictures. Because Logo's turtle graphics allows you to easily create images that imitate the pieces of plastic, it is possible to create quite attractive kaleidoscopic images on your computer screen with a simple set of procedures.

The Logo kaleidoscope operates in the following manner. The system contains a set of graphic procedures to draw the fundamental picture elements (squares, triangles, stars, etc.). There can be as many of these elements as you desire (subject to the memory limitations of your system, of course). Each of these elements can be drawn as large as you desire. This gives the effect of having even more patterns to choose from.

Next, we use Logo's random number generator to select a shape, a size for the shape, the shape's color, and a distance from the center of the screen at which the shape will be drawn. Finally, this data is used by another procedure that places a copy of the chosen shape at several equally spaced angles around the center of the screen. Once one shape has been drawn, the process can be repeated for other shapes until the final image meets with your approval.

The kaleidoscope we will demonstrate in this article is written in the MIT version of Logo for the Apple II and should work with most Logo systems with very few modifications.

The kaleidoscope was started out with six shapes.



The procedures for these shapes are:

```
TO TRI :SIZE
  LT 30
  REPEAT 3 [FD :SIZE RT 120]
  RT 30
END

TO DIAMOND :SIZE
  LT 45
  REPEAT 4 [FD :SIZE RT 90]
```

```
  RT 45
END

TO PATT1 :SIZE
  LT 30
  REPEAT 2 [FD :SIZE RT 60 FD :SIZE RT 120]
  RT 30
END

TO OCT :SIZE
  LT 67.5
  REPEAT 8 [FD :SIZE / 2 RT 45]
  RT 67.5
END

TO PATT2 :SIZE
  LT 60
  FD :SIZE RT 60 FD :SIZE RT 120
  FD :SIZE LT 60 FD :SIZE RT 120
  FD :SIZE RT 60 FD :SIZE RT 120
END

TO STAR :SIZE
  LT 18
  REPEAT 5 [FD :SIZE RT 144]
  RT 18
END
```

Each of these figures has been defined to have mirror symmetry on the vertical axis. This is not a requirement, and you may wish to experiment with other orientations. The octagon was drawn at half the specified size to keep it in balance with the other figures.

Constructing The Pattern

To make the kaleidoscopic image, we need a procedure that creates a list of basic patterns, chooses a pattern at random from this list, and selects an appropriate size (say between 20 and 50 units). Next, it should pick a random distance from the center (less than 60 units, to keep the images on the screen). Once these steps have been completed, copies of the chosen image should be stamped symmetrically around the screen. Then the procedure should wait for you to tell it if you want another element added to the image. When you press the RETURN key, the process will be repeated. The following procedure performs these tasks for us:

```
TO IMAGE
  MAKE "LIST [STAR DIAMOND OCT PATT1
    PATT2 TRI]
  MAKE "NAME SENTENCE PICKRANDOM :LIST
    ( 20 + RANDOM 30 )
  MAKE "DIST RANDOM 60
  PENCOLOR ( 1 + RANDOM 5 )
  PENUP
  WINDMILL :DIST :NAME
  MAKE "NAME REQUEST
  IMAGE
END
```

This procedure uses two other procedures that have to be defined: PICKRANDOM and WINDMILL. The function of PICKRANDOM is to choose an element of a list randomly. The following procedure does this for us:

```
TO PICKRANDOM :LIST
  OUTPUT PICK ( 1 + RANDOM (LENGTH :LIST ) )
    :LIST
END
```

The procedure PICK selects a given element from a list, and LENGTH measures the number of elements in a list:

```
TO PICK :NUM :LIST
  IF :NUM = 1 OUTPUT FIRST :LIST
  OUTPUT PICK ( :NUM - 1 ) ( BUTFIRST :LIST )
END

TO LENGTH :LIST
  IF :LIST = [] THEN OUTPUT 0
  OUTPUT 1 + LENGTH BUTFIRST :LIST
END
```

These two procedures operate "recursively." If you have a hard time understanding how they work, you may want to read about them in *Logo for the Apple II*, by H. Abelson, or read the chapter on recursion in my book *Discovering Apple Logo*. Also, we published some columns on recursion in "Friends of the Turtle" (COMPUTE!, November and December 1982).

Defining Windmill

The only procedure we have left to define is WINDMILL. The function of this procedure is to draw a chosen pattern at equally spaced angular increments around the center of the screen. You may want to experiment with different numbers of images. I have tried using six images spaced at 60-degree increments and eight images spaced at 45-degree increments. These both work fine, but other angles are worth exploring as well. The number of copies of a pattern times the angle increment must be 360 in order for the pattern to be symmetric. That is why we turn 60 degrees for 6 copies (6 x 60 = 360) and 45 degrees for 8 copies (8 x 45 = 360).

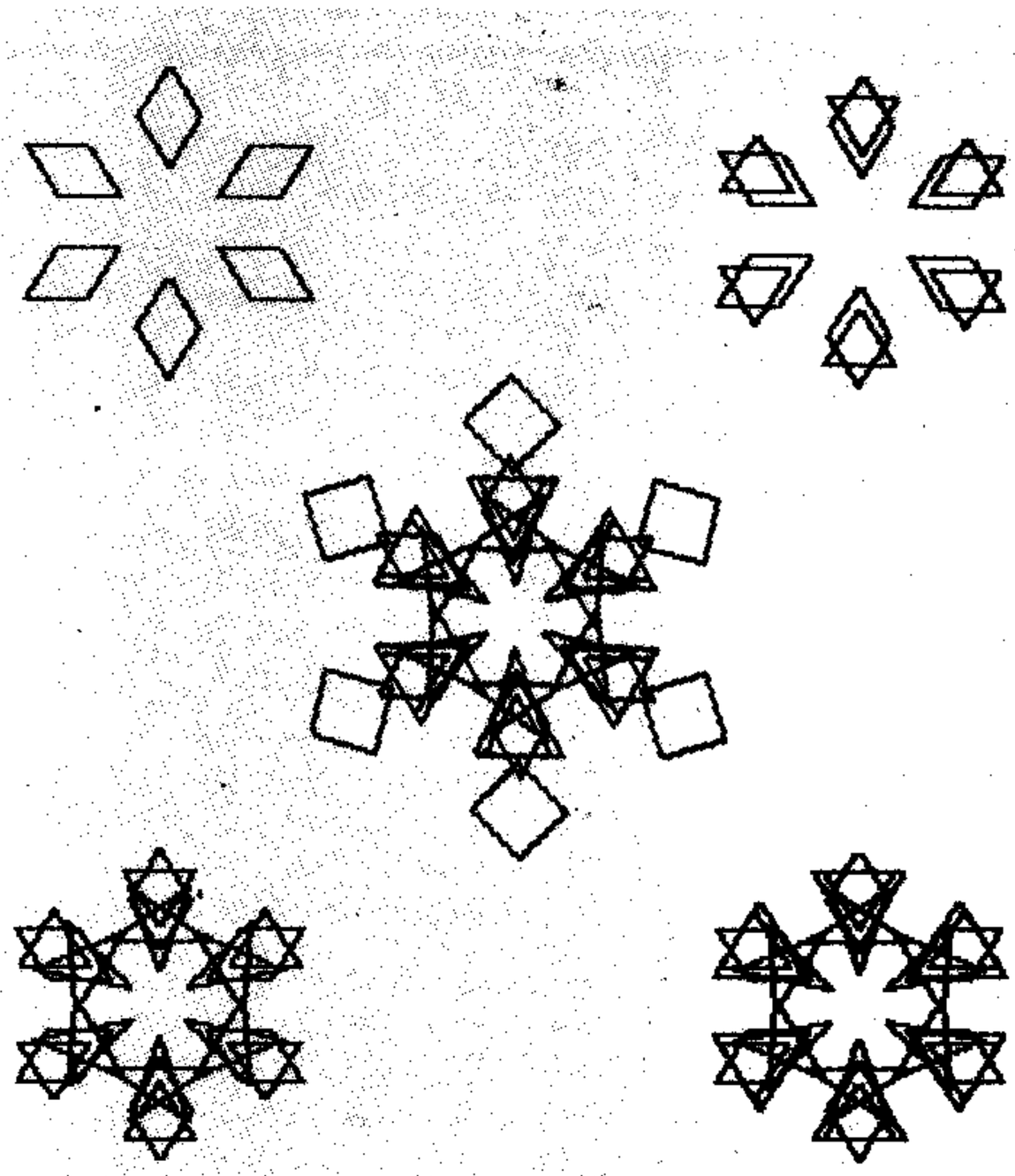
```
TO WINDMILL :DIST :LIST
  REPEAT 6 [FD :DIST PENDOWN RUN :LIST
    PENUP BACK :DIST RT 60]
END
```

To generate a kaleidoscopic pattern, hide the turtle and enter:

```
IMAGE
```

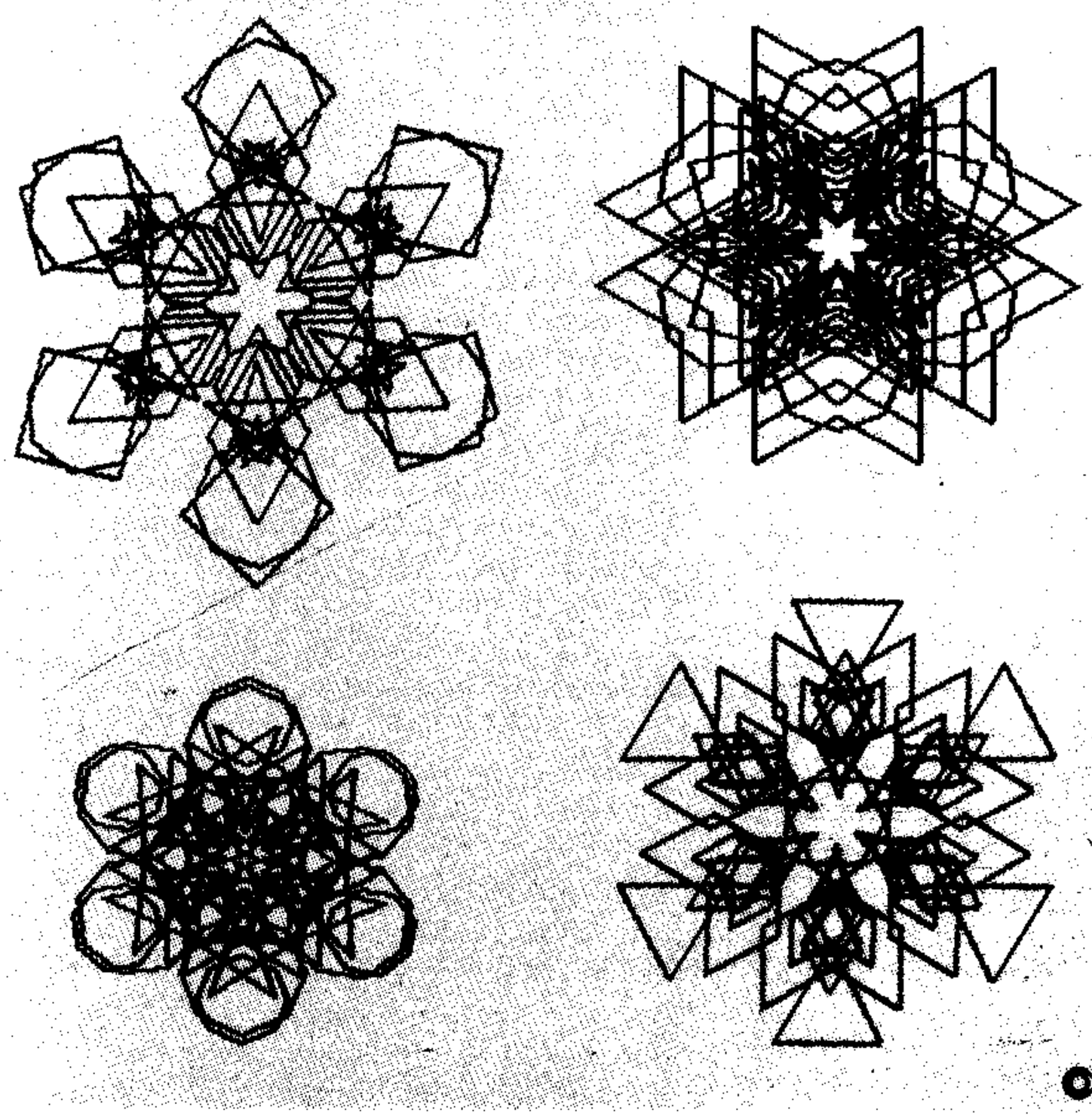
After the first pattern is drawn, press RETURN to get the next one. When the complexity of the pattern is satisfactory, you may want to print a copy of it or save it on your disk (with SAVEPICT, for example). If you are ambitious, you might want to write a Logo procedure that will keep track of all the randomly chosen values and generate its own Logo procedures for each pattern. Abelson's book (mentioned above) shows how to do this sort of thing.

The following five pictures show the successive development of one pattern:



The remaining figures illustrate some other kaleidoscopic patterns that were generated with this set of procedures.

I think you will agree that these patterns are more interesting than those created with colored dots.



Three Game Modules For The TI

Steve Davis

Last year, a young man named Michael Brouthers left his job at Texas Instruments in Dallas and boldly began a venture to develop game software for the TI home computer, a market that he felt was ready to blossom. When TI announced the $100 rebate on the 99/4A, the market for the machine did indeed grow rapidly.

Until now, Texas Instruments has been the only source of software packaged in the convenient Command Module, which TI invented for the 99/4. The module can contain ROM or GROM chips which contain a program (usually written in Assembler or GPL), and, in the case of TI's Mini-Memory Module, the cartridge can be used to add RAM to the console.

The main advantages to using program modules are:

- Ease of use. A person needs no peripheral devices or programming knowledge; just plug in the module and turn on the computer.
- Security. Programs cannot be copied or pirated easily since they reside in GROM or ROM chips. This also prevents accidental erasure of the program.
- Memory. An application program in a module takes up little or no console memory (RAM), so the computer's memory is available for data storage.

Using most third-party game software for the TI requires either Extended BASIC, Memory Expansion, Mini-Memory, Editor/Assembler, cassette or disk.

Now, Funware has introduced a line of game modules, *Henhouse*, *Rabbit Trail*, and *Video Vegas*, for the 99/4A. All use

the sprite graphics capability of the TI.

Henhouse

In *Henhouse*, you have five prolific chickens that lay eggs which roll down into five chutes. Each time a chute fills with eggs, you must take them to your truck without dropping them, all the while watching for wolves and poachers.

You get points for each poacher you shoot. Birds fly overhead, and you get points for shooting them, too. You play, using joysticks or the keyboard, until a wolf gets in the henhouse or you break six eggs.

The game may not seem as fast as some of the space or maze games in the arcades, but there are enough distractions that it requires concentration and the ability to do several things at once. It is simple enough to be enjoyed by users of all ages. The retail price is $39.95.

Rabbit Trail

This game is a cross between the *Donkey Kong* and *Frogger* type games. You are a hungry bunny who must hop along the trails and burrow through tunnels in search of carrots. You must not be eaten by a weasel or a hawk, be run over by a speeding car, or get caught in a trap.

Eating all the carrots without being caught advances you to the next level. You receive bonus points based on how fast you complete the level. If you are quick (as a rabbit should be), you may earn "bonus bunnies."

Each of the seven levels presents a more challenging screen. If you complete all seven screens, the game repeats from the first screen but with increased difficulty. Funware says that so far no one has been able to get higher than 24 screens, but to make it even that far would be an accomplishment.

Because of the graduated levels of difficulty, this game is suitable for both beginners and experienced game players. The keyboard may be used, but joysticks are recommended. The retail price for the module is $42.95.

Video Vegas

Anyone who has been to Las Vegas recently knows that some of the slot machines have been replaced by video versions. These operate like the mechanical ones except that the figures (bells, bars, cherries, lemons, etc.) are displayed on a video screen that simulates the rotating cylinders on a conventional slot machine.

Such is *Video Vegas*, a slot machine game that allows you to place $1, $2, or $3 bets by merely pressing keys on the computer console. This is not nearly as tiring as pulling those big levers in Vegas.

The color graphics of the figures are excellent; in fact, they look better than the graphics on some of the machines in Vegas and are a good example of the high-resolution pictures that can be drawn on the 99/4A.

There is nothing challenging about the module, which sells for $29.95, but people who like to play the slots will enjoy it.

Funware prefers that its modules be purchased from software dealers, rather than by mail order from the company.

Henhouse
Rabbit Trail
Video Vegas
Funware
405 N. Bowser
Building A
Richardson, TX 75081

C. Regena

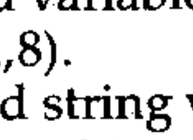
Subscripted Variables

TI BASIC allows variable names to be *subscripted*, or used in arrays of up to three dimensions. Examples of subscripted variables are A(1), ING$(2,6), and N(7,2,8).

Both numeric and string variables may use subscripts, which are written as numbers in parentheses after the variable name. The subscript itself may be a numeric variable or numeric expression. One constraint is that you cannot use the same variable name both with and without a subscript; that is, you cannot use the variable N and the variable N(3).

Just Like Mailboxes

I often think of variables as a mailbox system in memory:

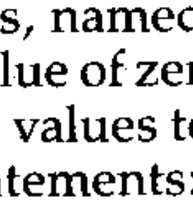


Here are two variables, named A and B. Initially, they each have the value of zero. As your program runs, you may assign values to these boxes. Suppose you have the statements:

```
100 B=7
150 A=A+1
```

The computer will put the value 7 in B's mailbox, then any later statement using B will simply use 7 in the formula instead of B. Line 150 says to add 1 to the value that is currently in A, then place the new value in A.

Some mailboxes are larger than others, and I compare these to subscripted variables. You might think of it as a big box for the Smith family – the first part of the box for John, the second part for James, and the third part for Jeremy. Here is our mailbox again:



The C box actually holds two values, which are written in TI BASIC as C(1) and C(2).

Boxes can be even larger – representing 1, 2, or 3 "dimensions," or using 1, 2, or 3 numbers in the subscripts. C(2) is the second element in the one-dimensional array of C above. N(2,4) would be an element in a two-dimensional array. X(3,4,2) would be an element in a three-dimensional array.

Arrays Are Workhorses

Arrays or subscripted variables can make a computer program more efficient in many cases. If you use a process several times, it may be worth using a variable with a subscript rather than several variables.

For example, suppose you are using your computer to sort a list of 25 students with their scores on a particular test. You could use the following method:

```
200 INPUT A$,A (FIRST STUDENT, SCORE)
210 INPUT B$,B (SECOND STUDENT, SCORE)
220 INPUT C$,C (THIRD STUDENT, SCORE)

ETC., FOR 25 STUDENTS

.
.
.
(SORT ROUTINE USING 25 VARIABLES)

.
.
.
600 PRINT A,A$
610 PRINT B,B$
620 PRINT C,C$

ETC., FOR 25 SORTED SCORES AND STUDENTS.
```

Using arrays or subscripted variables, you could INPUT the names as the N$ array and the

corresponding scores in the SC array, sort, and then print using this method:

```
200 FOR C=1 TO 25
210 INPUT N$(C),SC(C)
220 NEXT C
(SORT ROUTINE)
600 FOR C=1 TO 25
610 PRINT SC(C),N$(C)
620 NEXT C
```

Here is another example program that would be considerably longer if you did not use subscripted variables. Lines 110-130 READ from DATA a subject, a verb, and a phrase and put them in the S$, V$, and P$ arrays. Lines 140-190 contain the data (you could combine data lines if you wish). For the first time through the loop, S$(1) would be "I", V$(1) would be "RAN", and P$(1) would be "TO OUR HOUSE." S$(2) is "HE", V$(2) is "WALKED", and P$(2) is "TO THE STORE."; and so forth.

Line 200 uses the DEF function to define R6 as a random integer from 1 to 6. Each time R6 is used in the program, the computer will choose a random number from 1 to 6.

Line 210 clears the screen, and line 220 prints a title. Lines 230-240 choose a random S$, a random V$, and a random P$ to make up a sentence and print it. Line 250 returns to line 230 to repeat the process until you press CLEAR.

```
100 REM RANDOM SENTENCES
110 FOR C=1 TO 6
120 READ S$(C),V$(C),P$(C)
130 NEXT C
140 DATA I,RAN,TO OUR HOUSE.
150 DATA HE,WALKED,TO THE STORE.
160 DATA SHE,HOPPED,AROUND THE ROOM.
170 DATA IT,SPED,UP THE HILL.
180 DATA WE,ZOOMED,ACROSS THE GRASS.
190 DATA YOU,JUMPED,ALONG THE PATH.
200 DEF R6=INT(6*RND)+1
210 CALL CLEAR
220 PRINT "** RANDOM SENTENCES **":::
230 RANDOMIZE
240 PRINT :S$(R6);" ";V$(R6);" ";P$(R6)
250 GOTO 230
260 END
```

Memory Reserved

As soon as you specify a variable name with a subscript, the computer automatically reserves memory for an array with that name. If you use a variable D(3), the computer will automatically reserve elements up to D(10). In two-dimensional arrays, the computer will reserve up to N(10,10); and in three-dimensional arrays, the computer will reserve up to X(10,10,10).

If you need more than ten elements, use a DIMension statement to clear enough space. For example, for our 25 students and 25 scores in the program discussed previously, we would need a DIMension statement:

```
100 DIM N$(25),SC(25)
```

If your program is running nearly full memory and you do not need all the elements automatically reserved, you may save memory by dimensioning the array for the exact number you need:

```
100 DIM N(6)
```

The DIMension statement must appear before any reference to the array. I usually put my DIMension statements near the beginning of the program. You may specify several variables in one DIMension statement.

The computer actually starts all subscripts with the zero element, N(0). Thus, the automatic dimensioning includes 11 elements in arrays. If you prefer to use only elements numbered 1 and above, you may use the OPTION BASE statement to avoid reserving space for the zero elements:

```
100 OPTION BASE 1
110 DIM D(25,6)
```

Note: The OPTION BASE 1 statement must precede the DIM statement.

Combining The Ingredients

Following is an educational program which illustrates the use of subscripted variables. The program prints a recipe conversion problem for a math competency test. First, one of three recipes is printed. A random ingredient is chosen, and a random multiplication factor is chosen to print the problem. The student must choose from four possible answers.

Line 140 DIMensions the R$ array and the R array so the first subscript may go up to 3 and the second subscript may go up to 6. The first subscript will actually be 1, 2, or 3, which will correspond to the first, second, or third recipe. R$(C,0) will contain the title of the recipe for each of the three recipes. R(C,0) will be the number of servings each of the three recipes will make. R(C,I) and R$(C,I) contain the amount and the ingredient, where C is the recipe number and I is from 1 to 6. The values are read in as DATA in lines 150-230.

Lines 410-440 define values for the elements of the J array. These elements are multiplication factors for the conversion problem. These variables are used first to choose a factor for the problem, then to calculate the multiple-choice answers.

Program Structure

Lines	
100-130	Print title screen.
140	DIMension arrays for recipe elements.
150-200	READ from DATA the values for the R$ and R arrays.
210-230	DATA for recipes (please be careful while copying these lines – watch the commas and decimals).
240	Branch around subroutines.
250-390	Subroutines to convert decimals to fractions for printing the recipes and the multiple-choice answers.
400	Clear screen for problem.
410-440	Define multiplication factors.

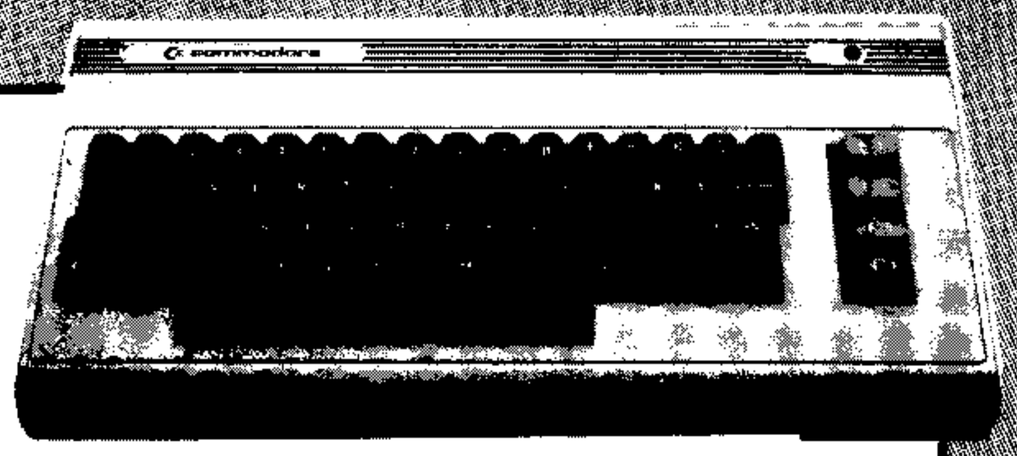
450-460 Randomly choose Recipe 1, Recipe 2, or Recipe 3. The variable C refers to the recipe number.
470-480 Print title of recipe and number of servings.
490-530 Print amount, measure, and ingredient six times. One of the recipes contains only five ingredients, so line 500 checks for a zero value. Line 510 converts the amount from a decimal to a fraction if necessary.
540-560 Randomly choose a multiplication factor for the problem. If F = 1 then J(1) = 1 which indicates no recipe conversion, and another number is chosen.
570-590 Draw a horizontal line of a random color under the given recipe.
600-640 Print the question, where A is the randomly chosen ingredient.
650 Calculate correct answer as N1.
660-750 Randomly print multiple-choice answers.
760-780 Sound a "beep" then wait for answer.
790-820 If answer is incorrect, play "uh-oh" and return for another answer.
830-870 Indicate correct answer and play arpeggio.
880-910 Print option to try another problem and branch appropriately.
920-930 Clear screen and END.

Math Competency Recipe Conversion

```
100 CALL CLEAR
110 PRINT TAB(6);"MATH COMPETENCY"
120 PRINT :::TAB(5);"RECIPE CONVERS
    ION"
130 PRINT
140 DIM R$(3,6),R(3,6)
150 FOR C=1 TO 3
160 READ R$(C,0),R(C,0)
170 FOR I=1 TO 6
180 READ R(C,I),R$(C,I)
190 NEXT I
200 NEXT C
210 DATA CHEESE SOUFFLE,2,2,TBSP BU
    TTER,2,TBSP FLOUR,1,C. MILK,.75
    ,C. GRATED CHEESE,2,EGGS,.5,TSP
     SALT
220 DATA DUMPLINGS,4,1,C. FLOUR,2,T
    SP BAKING POWDER,.5,TSP SALT,.5
    ,C. MILK,2,TBSP SALAD OIL,0,""
230 DATA PRONTO PUPS,6,2,EGGS,.5,C.
     MILK,.75,C. FLOUR,1,TSP BAKING
     POWDER,1,TSP SALT,.5,C. CORN M
    EAL
240 GOTO 400
250 N=R(C,I)
260 IF N<1 THEN 290
270 N$=STR$(N)
280 RETURN
290 IF N<>.75 THEN 320
300 N$="3/4"
310 RETURN
320 IF N<>.5 THEN 350
330 N$="1/2"
340 RETURN
350 IF N<>.375 THEN 380
360 N$="3/8"
370 RETURN
380 N$="1/4"
390 RETURN
400 CALL CLEAR
410 J(0)=.5
420 J(1)=1
430 J(2)=2
440 J(3)=4
450 RANDOMIZE
460 C=INT(RND*3)+1
470 PRINT TAB(7);R$(C,0)
480 PRINT :"SERVES";R(C,0)::
490 FOR I=1 TO 6
500 IF R(C,I)=0 THEN 530
510 GOSUB 250
520 PRINT N$;TAB(5);R$(C,I)
530 NEXT I
540 F=INT(RND*4)
550 IF F=1 THEN 540
560 F=J(F)
570 H=INT(RND*12)+5
580 CALL COLOR(13,H,H)
590 CALL HCHAR(24,1,128,32)
600 PRINT :::"IF YOU WANTED TO MAKE
    "
610 PRINT R$(C,0);" TO SERVE";F*R(C
    ,0)
620 A=INT(RND*5)+1
630 PRINT "HOW MANY ";R$(C,A)
640 PRINT "WOULD YOU NEED?"::
650 N1=F*R(C,A)
660 FOR CH=1 TO 4
670 X=INT(RND*4)
680 IF J(X)=-1 THEN 670
690 N=J(X)*R(C,A)
700 IF N1<>N THEN 720
710 ANS=CH
720 GOSUB 260
730 PRINT TAB(6);CHR$(64+CH);" "&N$
740 J(X)=-1
750 NEXT CH
760 CALL SOUND(150,1497,2)
770 CALL KEY(0,K,S)
780 IF S<1 THEN 770
790 IF K=ANS+64 THEN 830
800 CALL SOUND(100,330,2)
810 CALL SOUND(100,262,2)
820 GOTO 770
830 CALL HCHAR(19+ANS,7,42)
840 CALL SOUND(100,262,2)
850 CALL SOUND(100,330,2)
860 CALL SOUND(100,392,2)
870 CALL SOUND(200,523,2)
880 PRINT :"ANOTHER PROBLEM? (Y/N)"
    ;
890 CALL KEY(0,K,S)
900 IF K=89 THEN 400
910 IF K<>78 THEN 890
920 CALL CLEAR
930 END
```

SAVE

commodore VIC 64 $229.00



VIC 20 SPECIAL $89.00 INCLUDES GORTEK & MICROCHIPS VL110 Cassette program (retail value $19.95).

Commodore VIC20/64 1541 Single Disk Drive. . . $239.00
Commodore VIC20/64 1530 Datassette. 65.00
Commodore VIC20/64 1525 Printer. 229.00
Commodore VIC20/64 1701 Color Monitor. 249.00
Commodore VIC20/64 1600 Vicmoden (Telephone Interface). 55.00

Everything you need to support your COMMODORE VIC20 AND VIC64

Unbelievable prices for game and business software for your VIC20 and VIC64.

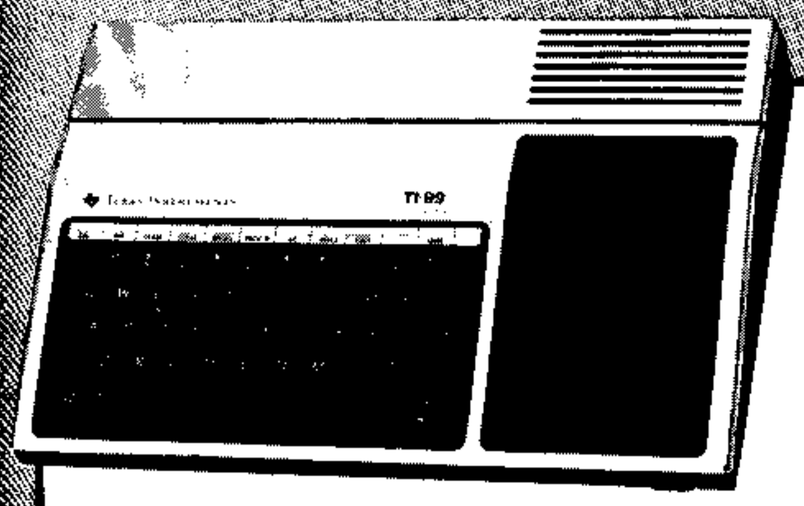
. . . . TOO MANY TO LIST BUT WE GOT IT . . . .

Micro Software International Inc

PractiCalc 20
For the Commodore VIC-20—
16K RAM required
Tape (CV2T10). $28.00
Disk (CV2D10). 33.00

PractiCalc 64 For the Commodore 64
Tape (C60T11). $37.00
Disk (C60D11). 39.00

Cardco, Creative Software, EPYX, HES, UMI, WICO, Romox.



Texas Instruments TI-99/4A Home Computer

ONLY $109.00*

*After TI manufacturer's rebate of $50.00.

SAVE

MORE GOOD DEALS

PHA 4100 Color Monitor. $349.95
PHP 2400 TI Impact Printer. 540.95
PHP 1100 Wired Remote Controllers. 27.95
PHP 2700 Program Recorder. 49.95

FREE EXPANSION BOX ($183. VALUE) with the purchase of any three of the following low price items:

PHP 1220 RS 232 Card. $135.95
PHP 1240 Disk Controller Card. 189.95
PHP 1250 Expansion System Disk Drive. 300.95
PHP 1260 Memory Expansion Card 32K. 227.95
PHP 1270 P-Code Card. 189.95
PHM 3111 TI Writer. 75.95
PHM 3113 Microsoft Multiplan. 75.95

FREE SPEECH SYNTHESIZER when you buy six solid state software, command cartridges.

QUANTITIES ARE LIMITED!

We carry in stock all hardware and software for TI home computers.

PHM 3026 Extended Basic. . . . . $ 75.95
PHM 3035 Terminal Emulator II. 38.95
PHM 3058 Mini-Memory. 75.95
PHM 3055 Editor/Assembler. 37.95
PHM 3109 TI Logo II. 74.95
PHM 3112 Parsec. 30.95
PHM 3053 TI Invaders. 44.95
PHM 3041-T Adventure (Pirate Adv.). 30.95
PHM 3057 Munch Man. 30.95
PHM 3006 Home Financial Decisions. 23.95
PHM 3044 Personal Report Generator. 30.95
PHM 3002 Early Learning Fun. 22.95
PHM 3090-97 Miliken Home Math Series K-8th grade. 30.95
PHM 3059-62 Scholastic Spelling Level 3-6. . . . . 41.95
PHM 3046-48 Scott, Foresman, Reading Pkgs. 41.95
PHM 3064 Touch Typing Tutor. 30.95
PHM 7008 Speaking Scholastic Spelling. 165.95
PHM 3122 Plato Interpreter. 37.95
PHM 3083-88 Addison Wesley Computer Math Games I-VI. 30.95

SAVE

SOUTHERN AUDIO VIDEO ELECTRONICS, INC.
1782 Marietta Blvd., N.W., Atlanta, Georgia 30318

Order Toll Free 1-800-241-2682
In Georgia (404)-351-8469

Get the best prices on TI Hardware and software. For a complete listing of all SAVE's products, send $5.00 for our catalogue (refundable with your first order).

Enjoy the convenience of in-home shopping. Call our toll free number today for orders only.

Use your American Express, VISA, Mastercard, check or money order. Minimum order of $50. Outside continental U.S.A. minimum order $250 U.S. currency. Shipping and handling charges are extra. All prices are subject to change without notice. Allow 2-4 weeks for delivery. Prices good through November, 1983. *$50.00 Rebate expires January, 1984.

TI Cadette: Computer Aided Design

Bradley Rogers

This clever program should provide hours of amusement for children who enjoy creating pictures. Similar to coloring or cut-and-paste, the computer screen becomes a magic window allowing easy design, color selection, and erasure. Requires Extended BASIC and joysticks.

"Cadette" is for children. Based on a scaled-down version of CAD (the Computer Aided Design), it transforms your TV screen into an electronic easel on which children can "draw" tropical birds, planes, surreal landscapes, or any number of other fascinating pictures. Joysticks and fire buttons are used instead of conventional pens and brushes.

Using these simple instruments, children can create intricate designs from a basic stockpile of 16 different shapes. Each shape can assume five different colors chosen at the start of the program. Cadette calls upon the imagination, but does not require highly developed motor skills. Most children over five should be able to manage it nicely.

Cadette is simple to use, with only four basic activities required:

1. Choosing a page (screen) color;
2. Choosing five brush (shape) colors;
3. Moving joysticks to position the shapes or the eraser; and
4. Pressing fire buttons to print or to erase.

The process is the electronic equivalent to pasting cutouts on construction paper. However, the program involves considerably less frustration than conventional craft activities. It permits children to erase neatly or to change their minds at any point without having to start over with a clean sheet.

Running The Program

Once the RUN command has been entered, a brief message appears, instructing you to select a page color. The page in this case is, of course, the TV screen. Next, you are confronted by a display of 12 colors, each identified by a number from 1 to 12. From this menu you select a screen color by pressing the appropriate number key and then the ENTER key. If you enter anything other than numbers 1 to 12, the computer waits patiently for you to reconsider.

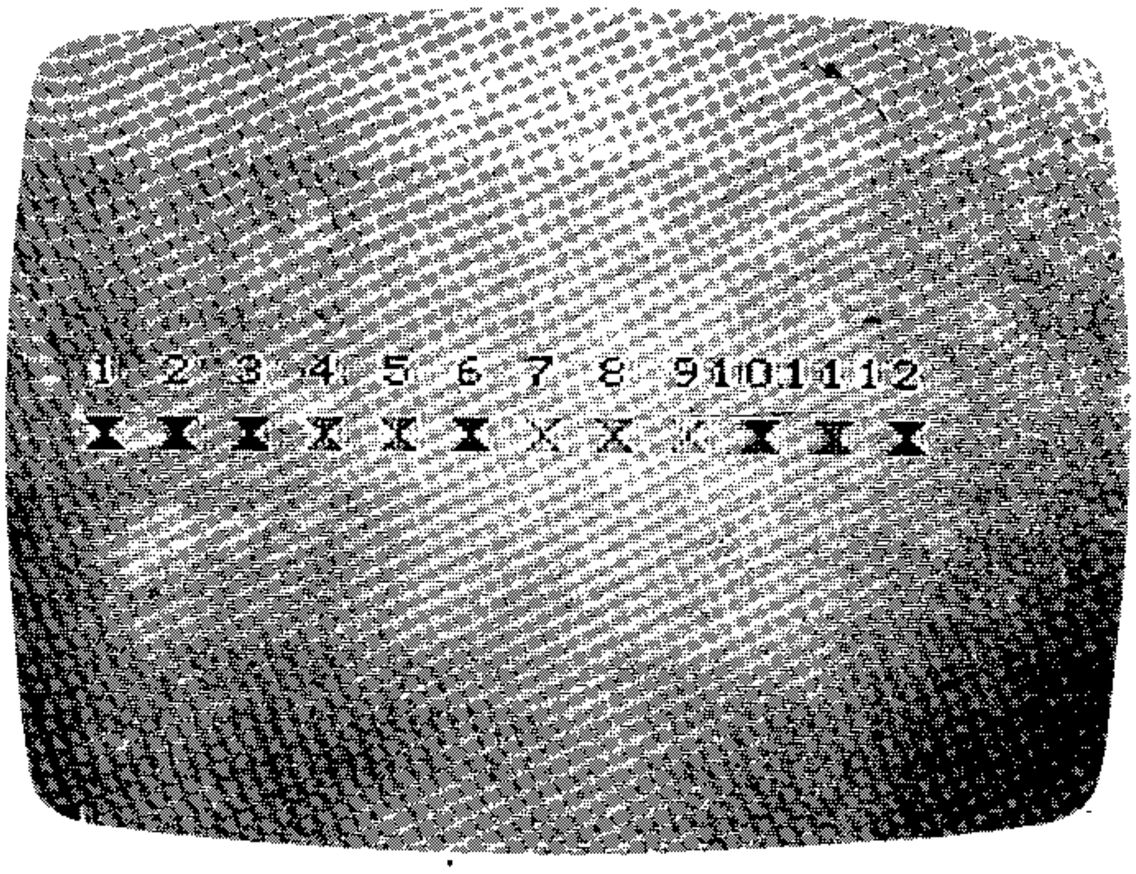
A second message now appears on the background color you chose. You are to select five brush colors. This message disappears, and you are asked to choose five from among twelve brush colors. The brushes in this case represent the colors of the shapes you will eventually use to create your design. Simply enter your five choices and remember to press ENTER after each selection.

After the color choices, the screen will blank and 16 geometric shapes will appear, eight across the top of the screen, and eight across the bottom. They consist of a circle, a square, assorted lines, triangles, and semicircles. Every few seconds the color of all 16 shapes changes, running through a cycle of five color changes, and then repeating.

Near the center of the screen is a small hollow box, which is the cursor. By using either of the joysticks, you move the cursor to capture and transport the colored shapes. After deciding which shape you want to capture, move the cursor to a position immediately adjacent to the shape. Once the shape turns the desired color, position the cursor on the shape.

The cursor will then disappear, and a duplicate of the colored shape you chose will appear immediately above or below the original, depending upon whether you selected from the top or bottom row. This duplicate may now be moved with the joysticks to any desired location. It will maintain its shape and color no matter what else happens on the screen. The original from which it was copied will remain in its display row and continue to undergo color transformations.

The duplicate shape, which now represents the cursor, can be placed at any position on the screen. Move it to the location you want and simply press the fire button. You will hear a low tone indicating that the button has done its job. If you have picked up the right joystick, the shape will "lock" at that screen location. Even if you move the cursor, the shape will remain fixed as



Twelve page and twelve brush colors are available in "Cadette" from the TI-99.

long as the program runs. If you have picked up the wrong joystick, the shape will be erased.

Assuming you have the "lock" joystick, you now have two options. You can move the cursor shape to a new position and print it again, or you can select another shape of the same color or the same shape of another color. If you choose a new shape, repeat the initial capture procedure. Remember, however, that the cursor no longer appears as a hollow box, but in the shape of your previous selection. But once it is placed on a new colored shape, it will automatically assume the new shape and color.

The "lock" joystick locks your selection at the location you want. The other joystick also controls the cursor, but is used to erase. To erase a "locked" shape, simply move the cursor on top of that shape and press the fire button. A higher tone will sound, the shape will disappear, and you can make another selection. To avoid confusion, you might label one joystick "lock" or "print" and the other "erase."

Extensions And Modifications

If you want to alter the shapes, you can change lines 540, 560, 580, and 600, which are DATA statements that contain the hexadecimal representations of the shapes. Each shape is defined by a string of 16 hexadecimal numbers.

Some children may find that the cursor moves too quickly, rushing past the space in which they wanted to print a shape. You can change the cursor's speed in line 920 by adjusting the limit (4) in the FOR/NEXT loop.

One interesting modification to the program would make it more versatile without requiring a great deal of extra programming. For example, a larger menu of shapes could be shown initially, and 16 could then be chosen from it. This would not be a terribly complicated program adjustment as long as you remember that the shapes must be read into S$. It is better to present the shape menu before the color menus; once you start fooling with color statements, all kinds of unexpected complications develop. In considering such modifications, just remember that often there is a trade-off between versatility and user-convenience. The program could become less fun to use if a child has to make too many decisions.

Cadette

```
100 CALL CLEAR
110 CALL SCREEN(15)
120 DISPLAY AT(5,6):"SELECT PAGE CO
    LOR, 1 TO 12."
130 FOR I=1 TO 800 :: NEXT I :: CAL
    L CLEAR
140 DIM Z(5):: DIM S$(16)
150 A$="3C7EFFFFFFFF7E3C" :: X=4
160 FOR I=62 TO 142 STEP 8 :: CALL
    CHAR(I,A$):: CALL COLOR(X,X-1,1
    ):: X=X+1 :: NEXT I
170 CALL CHAR(40,A$):: CALL COLOR(2
    ,16,1):: CALL COLOR(9,14,1):: C
    ALL VCHAR(12,5,40):: X=3
180 FOR I=62 TO 142 STEP 8 :: CALL
    VCHAR(12,X*2+1,I):: X=X+1 :: NE
    XT I
190 CALL CHARPAT(56,Z$):: CALL CHAR
    PAT(57,W$):: CALL CHAR(33,Z$)::
     CALL CHAR(34,W$)
200 FOR I=1 TO 12
210 IF I=8 THEN CALL VCHAR(10,19,33
    ):: GOTO 240
220 IF I=9 THEN CALL VCHAR(10,21,34
    ):: GO TO 240
230 DISPLAY AT(10,I*2):USING "##":I
240 NEXT I
250 ACCEPT AT(24,1)VALIDATE(DIGIT)B
    EEP:Y
260 IF Y<1 OR Y>12 THEN 250
270 IF Y=7 THEN Y=13
280 IF Y=1 THEN Y=15
290 CALL SCREEN(Y+1):: CALL CLEAR
300 FOR I=1 TO 14 :: CALL COLOR(I,2
    ,1):: NEXT I :: DISPLAY AT(6,4)
    :"SELECT 5 BRUSH COLORS,(6 SPACE
    S)1 TO 12."
310 FOR I=1 TO 800 :: NEXT I :: CAL
    L CLEAR
320 FOR I=4 TO 14 :: CALL COLOR(I,I
    -1,1):: NEXT I :: CALL COLOR(2,
    16,1):: CALL COLOR(9,14,1)
330 A$="FF7E3C18183C7EFF"
340 FOR I=62 TO 142 STEP 8 :: CALL
    CHAR(I,A$):: NEXT I :: CALL CHA
    R(40,A$)
350 IF Y=15 THEN Y=0
360 IF Y=13 THEN Y=7
370 CALL COLOR(Y+2,2,1):: X=6 :: CA
    LL VCHAR(12,4,40)
380 FOR I=62 TO 142 STEP 8 :: CALL
    VCHAR(12,X,I):: X=X+2 :: NEXT I
390 FOR I=1 TO 12
400 IF I=8 THEN CALL VCHAR(10,18,33
    ):: GOTO 430
410 IF I=9 THEN CALL VCHAR(10,20,34
    ):: GOTO 430
420 DISPLAY AT(10,I*2-1):USING "##"
    :I
```

```
430 NEXT I
440 FOR I=1 TO 5
450 ACCEPT AT(24,1)VALIDATE(DIGIT)B
    EEP:Z(I)
460 IF Z(I)<1 OR Z(I)>12 THEN 450
470 IF Z(I)=1 THEN CALL VCHAR(18,I*
    2+2,40)ELSE 490
480 GO TO 500
490 CALL VCHAR(18,I*2+2,46+8*Z(I))
500 NEXT I
510 FOR I=1 TO 500 :: AA=8*8 :: NEX
    T I
520 CALL CLEAR
530 FOR I=1 TO 16 :: READ S$(I):: N
    EXT I
540 DATA 187E7EFFFF7E7E18,0107070F0
    F070701,80E0E0F0F0E0E080,FF7E7E
    1800000000
560 DATA 0000000187E7EFF,FFFFFFFF
    FFFFFFFF,FFFEFCF8F0E0C080,000103
    070F1F3F7F
580 DATA 80C0E0F0F8FCFEFF,7F3F1F0F0
    7030100,8080808080808080,010101
    0101010101
600 DATA FF0000000000000,000000000
    00000FF,8040201008040201,010204
    0810204080
610 X=40
620 FOR I=1 TO 5
630 P=Z(I)+1
640 IF Z(I)=1 AND Y=0 THEN CALL COL
    OR(I*2,2,1):: CALL COLOR(I*2+1,
    2,1):: GO TO 690
650 IF Z(I)>1 AND Z(I)=Y THEN CALL
    COLOR(I*2,2,1):: CALL COLOR(I*2
    +1,2,1):: GO TO 690
660 IF Z(I)=1 AND Y>0 THEN P=16
670 IF Z(I)=7 AND Z(I)<>Y THEN P=14
680 CALL COLOR(I*2,P,1):: CALL COLO
    R(I*2+1,P,1)
690 FOR J=1 TO 16 :: CALL CHAR(X,S$
    (J)):: X=X+1 :: NEXT J
700 NEXT I
710 G2=32 :: H1=12 :: F1=16
720 CALL CHAR(37,"FF8181818181FF"
    ):: CALL VCHAR(H1,F1,37):: J=39
730 QW=1 :: UU=1
740 FOR I=1 TO 8 :: CALL VCHAR(24,I
    *2+4,J+I):: NEXT I
750 FOR I=9 TO 16 :: CALL VCHAR(1,I
    *2-12,J+I):: NEXT I
760 X=1
770 CALL JOYST(UU,F2,H2)
780 CALL KEY(UU,RV,SV)
790 IF (H1=1 AND H2=4)OR(H1=24 AND
    H2=-4)OR(F1=2 AND F2=-4)OR(F1=3
    0 AND F2=4)THEN F2=0 :: H2=0 ::
    GO TO 900
800 IF F2=0 AND H2=0 AND SV=0 THEN
    900
810 H3=H1-H2/4 :: F3=F1+F2/4
820 CALL GCHAR(H3,F3,G3):: CALL GCH
    AR(H1,F1,G1)
830 IF RV+QW=19 AND(H1=24 OR H1=1)T
    HEN 890
840 IF H3=24 AND G3<>32 THEN H3=23
    :: CALL VCHAR(H1,F1,32):: CALL
    VCHAR(H3,F3,G3):: G2=32 :: GO
    TO 870
850 IF H3=1 AND(G3<>32)THEN H3=2 ::
     CALL VCHAR(H1,F1,32):: CALL VC
    HAR(H3,F3,G3):: G2=32 :: GO TO
    870
860 IF (H3<>24 AND H3<>1)OR(H3=24 A
    ND G3=32)OR(H3=1 AND G3=32)THEN
     CALL VCHAR(H3,F3,G1):: CALL VC
     HAR(H1,F1,G2):: G2=G3
870 IF RV+QW=19 THEN CALL VCHAR(H3,
    F3,G1):: G2=G3 :: CALL SOUND(10
    0,110,2):: GO TO 890
880 IF RV+QW=17 THEN CALL VCHAR(H3,
    F3,37):: G2=32 :: CALL SOUND(10
    0,220,2)
890 H1=H3 :: F1=F3
900 QW=-QW
910 IF QW=1 THEN UU=1 ELSE UU=2
920 FOR AA=1 TO 4 :: NEXT AA
930 X=X+1 :: IF X=10 THEN 940 ELSE
    770
940 J=J+16
950 IF J>103 THEN J=39
960 GO TO 740
```

Sprite Editor For TI

Larry Long

Here's a way to get maximum use of sprites on the TI-99/4A – and a program that generates listings for your sprite creations.

A very powerful yet often unused feature of the TI-99/4A is its ability to display and control sprites. With the 99/4A and the Extended BASIC Module, it is possible to generate 28 sprites for display and independent simultaneous movement. Program 1 should convince any doubters that this can be done. Although a lot of colored letters floating around the screen are a bit pointless, if we can modify and control the sprites, we will have a most useful feature.

Sprites can be designed by drawing on a piece of graph paper and then converting the on/off pixels to a hexadecimal number. If the two largest sizes of sprites are used, the hexadecimal number describing the shape of the sprite would be 64 characters long. A solution is a sprite editor that will allow us to draw the pattern we want on the screen and then have the computer create the program we need to make that sprite pattern. Program 2 will do exactly that, and more. It will allow us to edit the sprite pattern. Then, when we press the L key, it will display a complete listing that would, if copied on paper and then entered into the computer, provide a sprite and the necessary routine to control its movement.

Your Options

When you run the program, the first display screen will be a design grid with a box-shaped cursor. The area under the cursor will initially be white (signifying an "off" pixel). Press 1 to change the color beneath the cursor to black (representing an "on" pixel) or to move the cursor about the grid with the arrow keys. To turn off a particular pixel, press 0 and the background color will be returned to white. When you have completed your design, press the P key to see it displayed as a sprite.

At this point, you are given several options. You can magnify your newly constructed sprite (M key), change its color (C key), change its background color (B key), or set it in motion (E, S, D, X keys). If you are not pleased with the sprite's shape, you can modify it by striking the T key or (if the changes required are quite drastic) simply press the A key to start with a fresh grid. On the other hand, if you are satisfied with your sprite and its color and directional parameters, press the L key to create the BASIC statements needed to achieve these effects.

If using the sprite editor is your only concern, then skip the rest of this article and go straight to Program 2 and enjoy this easy access to sprites.

How The Editor Works

To understand what makes the editor work, let's take a general overview of the program:

Lines	
100-260	Set up screen display.
270-460	Are the main loop of the designing portion of the program.
470-680	Evaluate the design, put its values in an array, read the values in the array, convert them to hexadecimal numbers, and then build a 64-character string to describe the sprite pattern.
690-770	Put the sprite on the screen and display new program instructions.
780-930	Main loop of the implementation portion of the program.
940-980	Change size of sprite.
1000-1150	Display a listing of the sprite program.
1160-1220	Change the color of the sprite and screen.

A cursor is needed to indicate where you are located on the design grid. I chose to use a sprite (line 220) because I could move it around freely without disturbing the display under it. Repositioning the cursor is accomplished in line 380 with a CALL LOCATE. The arrow keys reposition the cursor, and the ENTER key changes the area under the cursor.

What makes "Sprite Editor" so valuable is its ability to generate the hexadecimal pattern for the sprite. The loop from line 500 through line 560

determines the character in each position of the design grid and stores that value in the array B (R,C). Line 570 provides a string with all of the possible hexadecimal digits placed in ascending order. Line 580 sets M$ to be "null." The loop from line 590 to line 630 evaluates the array elements and converts each row in the left half of the design grid to a pair of hexadecimal digits and concatenates them to M$. Line 620 is probably the most significant line in this loop, as it provides the hexadecimal numbers. It causes the computer to look at a particular digit (element) in HEX$ determined by the values calculated for HIGH and LOW. Lines 630-680 perform the same operation as 590-630, only for the right half of the design grid.

Line 690 assigns the hexadecimal numbers to ASCII characters 104, 105, 106, and 107. It is necessary to specify only the first character number in the CALL CHAR statement. When this feature is used, it is required that you start with a character that is evenly divisible by 4. Line 730 actually displays the sprite.

Lines 740-770 provide instructions for the implementation portion of the program. Lines 780-830 check for specific key presses and provide appropriate branching to list the program; end the program; start from the beginning; change the background color; modify the existing sprite; change sprite size; or change sprite color. Lines 840-920 check for arrow key presses and then increment or decrement sprite speed.

Lines 940-980 change sprite size. Lines 1000-1150 display a program listing that would generate a sprite like the one designed by the Sprite Editor. One problem with listing the program is displaying the quote character. The computer interprets it to mean that you want to end the PRINT statement. The solution is to redefine an unused character (I chose the lowercase "n") to look like the quote character.

Finally, lines 1160-1220 allow you to change the color of the sprite and screen.

Program 1: Sprite Generation

```
100 CALL MAGNIFY(2)::FOR X=1 TO 28::
    CALL SPRITE(#X,64+X,X/2,96,128,I
    NT(RND*100)-50,INT(RND*100)-50):
    :NEXT X::GOTO 100
```

Program 2: Sprite Editor

```
100 REM SPRITE EDITOR
110 DIM B(16,16):: SC=1
130 C1=7
140 CALL CHAR(100,"")
150 CALL CHAR(101,"FFFFFFFFFFFFFFFF
    ")
160 CALL CHAR(102,"FFFFC3C3C3C3FFFF
    ")
170 CALL COLOR(9,2,16)
180 CALL CLEAR
190 DISPLAY AT(1,10):"SPRITE EDITOR"
200 FOR R=1 TO 16 :: CALL HCHAR(4+R
    ,2,100,16):: NEXT R
210 CALL MAGNIFY(1)
212 IF K=84 THEN GOTO 217
215 CALL SCREEN(8)
217 CALL DELSPRITE(ALL)
220 CALL SPRITE(#28,102,14,32,8)
225 CALL HCHAR(21,1,32,31):: CALL H
    CHAR(22,1,32,31)
230 DISPLAY AT(22,2):"E=UP X=DOWN S
    =LEFT D=RIGHT"
240 DISPLAY AT(23,2):"PRESS 1 - PIX
    EL ON ,0 - OFF"
250 DISPLAY AT(24,2):"PRESS P TO DI
    SPLAY SPRITE"
260 R=1 :: C=1
270 CALL KEY(0,K,S)
271 IF S=0 THEN 270
272 IF K=48 THEN KHAR=100
274 IF K=49 THEN KHAR=101
280 IF K=83 THEN C=C-1 :: GOTO 320
290 IF K=68 THEN C=C+1 :: GOTO 320
300 IF K=69 THEN R=R-1 :: GOTO 320
310 IF K=88 THEN R=R+1 :: GOTO 320
312 IF K=80 THEN 470
320 IF C<1 THEN C=16
330 IF C>16 THEN C=1
340 IF R<1 THEN R=16
350 IF R>16 THEN R=1
380 CALL LOCATE(#28,(8*R)+25,8*C+1)
420 CALL HCHAR(4+R,1+C,KHAR)
430 CALL SOUND(20,200,5)
460 GOTO 270
470 CALL DELSPRITE(ALL)
480 CALL HCHAR(21,1,32,128)
490 DISPLAY AT(22,2):"PLEASE WAIT W
    HILE I THINK."
500 FOR R=1 TO 16
510 FOR C=1 TO 16
520 CALL GCHAR(4+R,1+C,GC)
530 GC=GC-100
540 B(R,C)=GC
550 NEXT C
560 NEXT R
570 HEX$="0123456789ABCDEF"
580 M$=""
590 FOR R=1 TO 16
600 LOW=B(R,5)*8+B(R,6)*4+B(R,7)*2+
    B(R,8)+1
610 HIGH=B(R,1)*8+B(R,2)*4+B(R,3)*2
    +B(R,4)+1
620 M$=M$&SEG$(HEX$,HIGH,1)&SEG$(HE
    X$,LOW,1)
630 NEXT R
640 FOR R=1 TO 16
650 LOW=B(R,13)*8+B(R,14)*4+B(R,15)
    *2+B(R,16)+1
660 HIGH=B(R,9)*8+B(R,10)*4+B(R,11)
    *2+B(R,12)+1
670 M$=M$&SEG$(HEX$,HIGH,1)&SEG$(HE
    X$,LOW,1)
680 NEXT R
690 CALL CHAR(104,M$)
700 CALL MAGNIFY(3)
710 MM=3
720 M=4
730 CALL SPRITE(#1,104,C1,50,170,0,
    0)
740 DISPLAY AT(21,2):"C COLOR  M MA
    GNIFY  T EDIT"
750 DISPLAY AT(22,2):"A ERASE  Q QU
    IT  B BACKGRD"
```

```
760 DISPLAY AT(23,2):"E=UP X=DOWN S
    =LEFT D=RIGHT"
770 DISPLAY AT(24,8):"L LISTS PROGR
    AM"
780 CALL KEY(0,K,S)
790 IF K=76 THEN GOTO 1000
800 IF K=81 THEN GOTO 990
810 IF K=65 THEN GOTO 100
812 IF K=66 THEN GOSUB 1200
815 IF K=84 THEN GOTO 210
820 IF K=77 THEN GOTO 940
830 IF K=67 THEN GOTO 1160
840 IF K=83 THEN H=H-2
850 IF K=68 THEN H=H+2
860 IF K=69 THEN V=V-2
870 IF K=88 THEN V=V+2
880 IF V>120 THEN V=120
890 IF V<-120 THEN V=-120
900 IF H>120 THEN H=120
910 IF H<-120 THEN H=-120
920 CALL MOTION(#1,V,H)
930 GOTO 780
940 CALL MAGNIFY(M)
950 MM=M
960 IF M=3 THEN M=4 ELSE M=3
970 FOR D=1 TO 20 :: NEXT D
980 GOTO 780
990 STOP
1000 REM PROGRAM LISTER
1010 CALL CHAR(110,"002424")
1020 CALL CLEAR
1030 PRINT "{6 SPACES}PROGRAM LISTI
     NG"
1035 CALL DELSPRITE(ALL)
1040 PRINT
1050 PRINT ">100 CALL CHAR(104,n";:
     : FOR W=1 TO 64 :: PRINT SEG$(
     M$,W,1);:: NEXT W :: PRINT "n)"
1055 PRINT ">105 CALL SCREEN(";SC;"
     )"
1060 PRINT ">110 CALL MAGNIFY(";MM;
     ")"
1070 PRINT ">120 CALL SPRITE(#1,104
     ,";C1;",150,150,";V;",";H;")"
1080 PRINT ">130 CALL KEY(0,K,S)"
1090 PRINT ">140 IF K=68 THEN H=H+2
     "
1100 PRINT ">150 IF K=83 THEN H=H-2
     "
1110 PRINT ">160 IF K=88 THEN V=V+2
     "
1120 PRINT ">170 IF K=69 THEN V=V-2
     "
1130 PRINT ">180 CALL MOTION(#1,V,H
     )"
1140 PRINT ">190 GOTO 130"
1150 PRINT :: PRINT :: PRINT :: PRI
     NT :: PRINT
1155 DISPLAY AT(21,3):"A - ERASE
     {3 SPACES}Q - QUIT"
1156 CALL KEY(0,K,ST):: IF ST=0 THE
     N 1156
1157 IF K=81 THEN GOTO 990
1158 IF K=65 THEN GOTO 100
1159 GOTO 1156
1160 C1=C1+1 :: IF C1>16 THEN 1180
1170 CALL COLOR(#1,C1):: GOTO 780
1180 C1=2 :: CALL COLOR(#1,C1):: GO
     TO 780
1200 REM SCREEN COLOR CHANGE
1210 SC=SC+1 :: IF SC=17 THEN SC=2
1220 CALL SCREEN(SC):: RETURN
```

COMPUTE!'s Programmer's Reference Guide to the TI-99/4A

Author: C. Regena
Price: $14.95
On Sale: Now

Just about the best way to learn how to program a computer is to sit down with a patient friend who already knows how, and ask questions while you experiment with the computer. Owners of the popular Texas Instruments home computer will find that C. Regena is that kind of friend, and *Programmer's Reference Guide to the TI-99/4A* is that kind of book.

Regena carefully explains every BASIC command and function, and all the techniques needed to program TI graphics, sound, and speech. It's hard to think of a question that she doesn't answer simply and clearly, with hints about ways to write programs that do exactly what you want.

The book also provides dozens and dozens of programs, ranging from very short examples to full-length commercial-quality software. In effect, readers can look over Regena's shoulder as she goes through the programming process step by step, explaining what she's doing as she goes along. Not to mention the fact that the finished programs are valuable in their own right.

Even readers who are familiar with the computer will find this book valuable as a reference, where they can look up information they need and find the answers to particular questions.

Above all, *Programmer's Reference Guide to the TI-99/4A* is a book that lets readers use it however they like. You don't have to start at page one and read through, following someone else's plan for what you should learn first and what can wait until later. Instead, you can explore this book from any point of view, to solve almost any programming problem, and find the answer quickly and easily.

C. Regena is **COMPUTE!** Magazine's regular columnist on the TI-99/4A. She's an experienced and resourceful programmer. Like most of her readers, she taught herself how to program, and she hasn't forgotten what it's like to be a beginner, just starting out with the computer. And with *Programmer's Reference Guide*, TI users now have Regena to help them learn how to make their computer do exactly what they want it to do.

Programmer's Reference Guide to the TI-99/4A is available from **COMPUTE!** Publications, the leading publisher of books and magazines for home, educational, and recreational computing.

Available at computer dealers and bookstores nationwide. To order directly call TOLL FREE 800-334-0868. In North Carolina call 919-275-9809. Or send check or money order to **COMPUTE! Books**, P.O. Box 5406, Greensboro, NC 27403.

Add $2 shipping and handling. Outside the U.S. add $5 for air mail, $2 for surface mail. All orders prepaid, U.S. funds only.

Table of Contents

Timex Tutorial

The Programming Kit I, a BASIC tutorial program for Timex/Sinclair users, is among three new programs produced by Timeworks. *The Programming Kit*, a how-to learning approach to programming, includes an explanation of an eight-step approach to program design.

Family Pak is a set of five 2K programs designed for day-to-day home tasks. The programs are Check Book Balancer, Recipe Recorder, Mini-Money Manager, Homework Helper and Memoboard, a family message center.

Timeworks' third new program, *Scyon's Revenge*, is a deep-space combat game that includes 3-D simulation. The game is provided with a keyboard overlay to give you the feeling of punching command buttons rather than hunting and pecking on a keyboard.

Timeworks, Inc.
405 Lake Cook Road
Building A
Deerfield, IL 60015
(312)291-9200

Kindergarten Gallery

Midwest Software has developed a series of computer programs designed for kindergarten children.

The *Kinder Koncepts* software addresses reading awareness, math concepts, pattern recognition, letters, numbers, colors, and shapes.

All programs follow the same general format, operate with a single keystroke, and keep the necessity for reading to a minimum.

In each program, ten problems are presented. A correct answer is rewarded with a smiling face and a tune. An incorrect answer results in a frown and the chance to try again. Each program has a built-in graph so that progress can be monitored at a glance.

The programs are available on cassette or disk for all Commodore computers except the VIC-20. Disk versions also are available for the Apple II+. The cost is $7.95 per program for cassette or $69.50 for a disk with ten programs.

Midwest Software
Box 214
Farmington, MI 48024
(313)477-0897

Mickey's New Adventure

Walt Disney Productions has entered the computer software market, and it's making its debut with the help of Mickey Mouse.

Mickey in the Great Outdoors is a pair of interactive adventure games for children seven to ten years old. Mickey Goes Hiking develops grammar and spelling skills by requiring players to finish sentences and unscramble words to help Mickey through his adventure. Mickey Goes Exploring is a similar game, but is based on math skills and equation solving.

Mickey in the Great Outdoors, is being offered only for Atari computers, and distributed through Atari. This program, however, is just the tip of Walt Disney's software iceberg, according to the company.

Plans call for as many as 50 additional Walt Disney programs to be released this year, sup-

Mickey Mouse helps unscramble a word in Mickey in the Great Outdoors.

porting Atari, Radio Shack, Texas Instruments, NEC, and Panasonic computers, and distributed through those companies.

Walt Disney Telecommunications
500 South Buena Vista St.
Burbank, CA 91521
(213)840-1111

Add 64K To Timex/Sinclair

Sunflower Systems has produced a 64K RAM pack for the Timex/Sinclair computers.

The self-contained memory expander plugs into the computer just like the Timex 16K RAM. No additional equipment is needed.

The unit, housed in metal to eliminate radio frequency interference, sells for $119.95 plus $5 for shipping.

Sunflower Systems
718 East Avenue B
Hutchinson, KS 67501
(316)662-2134

Science Fiction Text Adventure

Cyborg is a science fiction text adventure that has no treasures to collect and no score to tally.

The adventure is available from Sentient Software for $34.95 in versions for Atari, Commodore 64, Apple, and IBM.

The game includes character development, animals to talk to, opinions from the Cyborg, and a command structure that allows full sentences.

Sentient Software, Inc.
P.O. Box 4929
Aspen, CO 81612
(303) 925-9293

Turn On The Juice

Tronix, a young company that made its first splash in the VIC-20 market, has added the Commodore 64 to its repertoire.

The company's latest creation is *Juice*, a fast-paced strategy game for the 64 and Atari computers. The hero in *Juice* is Edison, whose job is to complete circuit boards in the face of all the troubles his adversary – Killerwatt – can throw his way.

The game includes six play levels, each with three rounds plus a bonus round. The 32K Atari version sells for $29.95, and the Commodore 64 version sells for $34.95.

Another Tronix offering for the 64 is *Kid Grid*, which previously had been released in an Atari version. In the game, "the Kid" darts around a grid trying to connect all the dots while eluding four bullies. *Kid Grid* sells for $34.95.

In addition to branching into the 64 market, Tronix has bolstered its VIC-20 lineup with the addition of three new cartridge games, *Deadly Skies*, *Scorpion*, and *Gold Fever!*.

Deadly Skies is a shoot-em-up

game in which the player, equipped with a squadron of helicopters, takes on a military base. *Gold Fever!* is a maze game in which a prospector faces runaway boxcars, boulders, claim jumpers, and a limited supply of oxygen. The object of *Scorpion* is to keep the snake alive and fed in the midst of a world filled with dragons, frogs, Venus's-flytraps, stalkers, worms, and pods.

Each of the VIC-20 games sells for $39.95.

Tronix Publishing, Inc.
8295 S. La Cienega
Inglewood, CA 90301
(213)215-0529

A Program To Remember

Memory Trainer, an interactive program to teach memory improvement, is available from Einstein.

The program, which is available for the Apple, Atari 800, and Commodore 64, is based on memory improvement research from the past 100 years.

MemoryTrainer includes five lessons in a three-disk package that sells for $89.95. The lessons teach the ability to remember faces, dates, telephone numbers, lists, and quotations, and to use association as a memory tool.

The package also includes Memory Mix, a game that provides practice for each memory skill.

The Einstein Corporation
11340 W. Olympic Blvd.
Los Angeles, CA 90064
(213) 477-6733

VIC Wafer Storage

A low-cost micro-wafer storage device for the VIC-20 will be available later this year from Unitronics, through a licensing agreement with Vadem, the unit's builder.

The V-20 Expander is described as an inexpensive alternative to floppy disk storage for low-end computers. It reads or writes data to small tape cassettes at a speed approaching that of disks.

The device, which measures 5x6x7 inches, plugs into the VIC's cartridge expansion slot. It includes a 10K RAM memory expansion board, a 64K data wafer and high-speed micro-wafer drive, a filing system, and VWOS – the Vadem Wafer Operating System.

Because VWOS is able to access the computer's memory bus directly, rather than through a serial port, the V-20 is able to improve on the data transfer rates of existing micro-wafer devices.

The expander is expected to sell for about $100.

Vadem
3517 Ryder Street
Santa Clara, CA 95051
(408) 738-0571

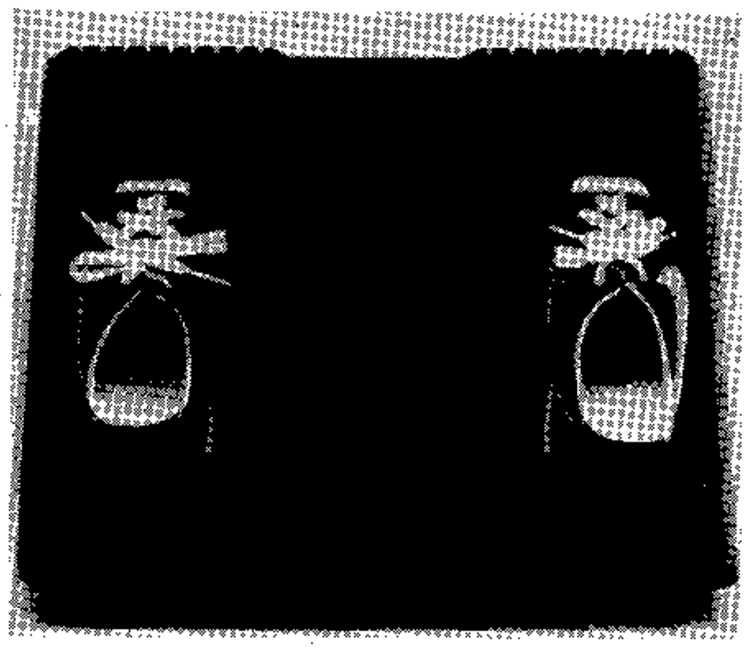
Stand-On Game Controller

The Joyboard, a game controller that involves the whole body rather than just the hands, has been introduced by Amiga for the VIC-20 and Atari computers.

The Joyboard, which comes with *Mogul Maniac*, a skiing simulation game, will sell for about $50. Other games designed for use with the Joyboard – *Surf's Up* and *Off Your Rocker* – will cost about $20.

The Joyboard also can be used with many existing maze-type games to provide a different challenge, or, for shoot-em-up games, a conventional joystick can be plugged into the Joyboard to control firing, while your feet control direction.

Amiga also has produced a version of its Power-Stick joystick for the TI-99/4A. This includes two controllers hard-



The Joyboard from Amiga transfers videogame control from your hands to your entire body.

wired into a single plug to fit the TI's single jack configuration. The pair will sell for about $20.

Amiga Corporation
3350 Scott Boulevard, Building 7
Santa Clara, CA 95051
(408) 748-0222

Add-On Adventure

Infocom, the company that produced the *Zork*, the popular underground fantasy series, is launching a new trilogy that will take the adventurer into the world of magical powers and perilous predicaments.

The first in the new series, *Enchanter*, scheduled to be available by mid-September, is a prose adventure that takes place in an abandoned castle. The passage of time plays an important role in the game: you must eat, drink, and sleep regularly, or your powers will fail.

The game, which will be available in versions for most popular microcomputers, will retail for $49.95 to $59.95.

Infocom, Inc.
55 Wheeler St.
Cambridge, MA 02138
(617) 492-1031

Battle Games

Thorn EMI Home Video has released a pair of new battle games, *Orc Attack* and *Fourth Encounter*.

In *Orc Attack*, the player must defend his castle against the Orcs, who erect ladders and scale the castle walls under cover of a volley of crossbow bolts from their archers. The game, which is available for the Atari 400 and 800, sells for $39.95.

Fourth Encounter is a cartridge game for the VIC-20. The challenge here is to save a planet from an invasion of aliens, who bring with them slavery, death, and destruction. *Fourth Encounter* is available for $39.95. In addition to these two new games, Thorn EMI has converted a couple of other games into new formats. *Submarine Commander*, previously released as an Atari game, is now available for the VIC, and *River Rescue* now can be played on the Atari.

Thorn EMI Home Video
1370 Avenue of the Americas
New York, NY 10019

APPLE / ATARI / COMMODORE

MEASURE & CONTROL TEMPERATURE

DISPLAY GRAPHICS
HARD COPY OUTPUT
ALARMS & SETPOINTS
DISK FILE DATA STORAGE

1–256 Sensors
Precise to 1/100 Degree
Complete Software
$129.00 Package

American Data Cable, Inc.
2864 Ray Lawyer Drive, #205-352
P.O. Box 2212 • Placerville, CA 95667
(916) 622-3465

PERSONAL PERIPHERAL PRODUCTS *presents:*

SPEAKEASY

VIC-20 SPEECH

COM-64 SPEECH

Cartridge, Instructions & Dictionary	$79.00
FALL SPECIAL: Cassette Editor & Extension Speaker, Reg. $19.95, with **SPEAKEASY**	**FREE**
Commodore 64 Adapter Board	$12.95

BARE BONES BOARDS

	Assembled	Kit
8K Ram/VIC-20 Block Switched	$39.00	$29.95
16K Ram Expander for VIC-20	$64.95	$49.95
3 Slot Switched & Fused Board/VIC-20	$34.95	$24.95
NEW: 4 Slot Switched & Fused Board/COM-64	$44.95	$34.95
COMING SOON: Mighty Modem VIC 20/COM-64	$79.95	

ADD $2.00 Total Order Handling/Ill.Residents Add 6% Sales Tax

PERSONAL PERIPHERAL PRODUCTS
P.O. BOX 3423
FOX VALLEY MALL
AURORA, IL 60505 • (312) 961-2347
COM-64 & VIC-20 IS A TRADEMARK OF COMMODORE

FREE!
VIC-20 COMMODORE 64
USERS GROUP MEMBERSHIPS

With software purchase. Send for your FREE DISCOUNT software catalogue today!
Group benefits include:
- Newsletter (full of programs, reviews & ideas)
- Access to extensive club library
- Questions hot-line
- Free ads for members
- Contests
- Discounts
- Software and hardware locator service
- Members only repair service
- AND MUCH MUCH MORE!!!

MEMBERSHIPS ALSO AVAILABLE WITHOUT SOFTWARE PURCHASE

Write or call today for FREE Catalogue
803/797-1533
LORDS OF BASIC
P.O. Box 459, Dept. 101
Ladson, SC 29456
Distributor & Dealer Inquiries Invited
VIC-20 • Commodore 64 are Reg. TM of CBM

ATTENTION
TEXAS INSTRUMENTS TI-99/4A
OWNERS

We have hundreds of 3rd party independent software programs on cassette and disk ready to run on your TI-99/4A. Games, business, and educational programming at discount prices as low as $8.95 ea. Plus all TI hardware and software at incredibly low, low prices, including the new TI-99/2 and CC-40 computers. We also have dust covers, heavy duty joysticks with TI adapters, and many more accessories. Call or write now for a FREE listing. We ship your order U.P.S. the same or next business day to insure fast service. Visa and MasterCard accepted (NO service charges) or C.O.D. is okay.

THE MUSIC WORKSHOP
59 E. Tioga St.
Tunkhannock, PA 18657
CALL 1-717-836-4522

SOFTWARE COMMODORE 64

The Staff: Polyphonic Music Editor & generator. Enter up to 93 measures of 3 part harmony on easy to use graphic display. Disk: $22.95 for 64, $17.95 for VIC. Tape: $19.95 for 64, $14.95 for VIC. Add $1.50 for Postage and Handling. Check or Visa, Master Card accepted.

We have a large selection of software for the 64—Word Processors, Data Base, Mailing List, Accounting Package, Spread Sheet, Educational Applications, Home and Personal Record Keeping, Programmers Aids, Games.

ASK FOR FREE CATALOG

PROFESSIONAL MICRO SERVICE

100 W. 22nd St., P.O.B. 7268

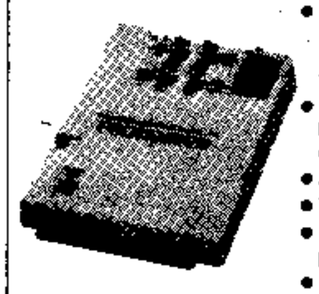
Baltimore, Md. 21218

301-366-0010

Dealer inquiries invited.

Commodore 64 is a registered trademark of Commodore Business Machines.

Wasting Money? We Have the World's Most Cost Effective Development System.



Send for Free Brochure

VISA AND MASTERCARD ACCEPTED

- Includes Hexkit 1.0, a powerful 100% machine code editor/debugger utility program that makes coding for 8-bit Micros a snap.
- Program from Commodore VIC-20 keyboard into built-in 4K ROM emulator
- Jumper to target ROM socket
- Test programs in circuit
- Built-in EPROM programmer and power supply
- Burns & runs EPROMS for the Commodore VIC-20, too
- Comprehensive manuals
- Fits EXPANSION PORT

PROMQUEEN CARTRIDGE ONLY $199 U.S. $269 CAN.

+ Prices	U.S.	Can.
Promqueen 64	$299.00	$399.00
8K board w/ 1 Eprom	$ 29.95	$ 39.95
16K board w/ 1 Eprom	$ 39.95	$ 49.95
8K ROM board w/ 1 Eprom-C64	$ 39.95	$ 49.95

Distributed in U.S. by **Arbutus Total Soft, Inc.**
4202 Meridian, Suite 214
Bellingham, WA 98226
Ph. 800-426-1253, in Washington 206-733-0404

Distributed in Canada by
IBC/Distribution Canada
4047 Cambie St., Vancouver, BC V5Z 2X9
Ph. 604-879-7812

VIC-20 VIC-20

Postage Paid

SOFTWARE

100% Machine Code
Money Back Guarantee

BOMBER RUN $13.95 U.S.
Destroy Enemy Base with bombs, shoot down fighters with air to air missiles but beware of suicide jets. Clear enough territory to land, watch out for those unexploded bombs. Great colour graphics and sound.

SUPAVADERS $13.95 U.S.
They're back. But this time you have a bigger territory to defend (screen scrolls). Their ships stay high, raining missiles, preparing the way for commando crews in landing craft. Fire in four directions.

MERLIN'S MAZE 3D $13.95 U.S.
Find the seven treasures of Camelot. Collect Merlin's spells to increase your chance of success. Only they're invisible so good luck, lift wall. But it costs you energy. Solve logic problems to continue to next section.

CHADWELL'S SOFTWARE
#203, 4144A 97 St., Edmonton
Alberta, Canada T6E 5Y6
U.S. ORDERS WELCOME
DEALER INQUIRIES WELCOME

BEAT THE HIGH COST OF SOFTWARE

Before you buy any more software, you owe it to yourself to join SOFTRADERS™ the fastest growing worldwide trading network. Members swap software and info for all computers.

Annual membership includes
- Quarterly trading directories
- Monthly and flash updates
- Personalized trader listings
- Trader support systems
- Plus much more

Imagine having access to hundreds even thousands of programs each costing no more than the price of a stamp or phone call. Join now to beat the price increase and find out about the new member $$ offer. $49.95 U.S. $64.95 FOREIGN SASE for info. Indicate computer model.

SOFTRADERS™ INTERNATIONAL
1610 Shomaker Dr.
Murphysboro, IL 62966

NEW

ADD **64** MORE **K**'S
TO YOUR PERSONAL COUMPUTER

GREAT GIFT **$3.95** GREAT GIFT

64 K KARD

COMPLETE WITH INSTALLATION DOCUMENTATION

PLEASE RUSH ME ____ COPY(S) OF THE 64K KARD AT $3.95 EACH

SPECIFY COMPUTER

TO ORDER SEND CHECK OR MONEY ORDER TO:

COMPUTER HUMOR
P.O. BOX 859
PARK CITY, UTAH 84060

CLIP CLIP

C93

DUST COVERS

For Personal Computers, Peripherals, Game Units – Protective, Long-Lasting Vinyl Resists Both Dust and Liquids.

— CHOICE OF COLORS —

Amdek	IBM PC
Apple	Mattel
Atari	Rana Systems
BMC	Sanyo
Commodore	Star Micronics
Coleco	TI 99/4
Epsom	Timex Sinclair
Franklin Ace	TRS 80

FOR ***FREE BROCHURE*** WRITE:

ENCHANTED FOREST
P.O. Box 5261, Newport Beach, CA 92662
(1129 W. Balboa Blvd)

COMMODOR 64

"ABC" THE ALPHABET FOR PRESCHOOLERS

TEACH YOUR CHILDREN THE ALPHABET EARLY WITH GRAPHICS AND SOUND

ONLY $18.95 ON CASSETTE
OR $21.95 ON DISK

VIC-20

(CASSETTES ONLY)

THE ALPHABET GAME .....$ 9.95 (ea)
LEARNING NUMBERS 0-50 PART I....$ 9.95 (ea)
LEARNING NUMBERS 51-100 PART II....$ 9.95 (ea)
PART I&II ARE AVAIL. IN: ENGLISH; SPANISH; FRENCH; AND GERMAN VERSIONS PLEASE STATE LANG
ANY 3 VIC-20 CASSETTES FOR ONLY $ 25.-
Check or Money order +$1.50 Post. N.Y.Res+7¼tx.

THOR SOFTWARE
47 CHERNUCHA AVE.
MERRICK, N.Y. 11566

1/2 PRICE INTRODUCTION
Commodore & TI

Spreadsheet	**19.95**
Data Base	**19.95**
Word Processor	**9.95**
Payroll	**9.95**
Mailing List	**9.95**
Inventory	**9.95**
Games Pak*	**19.95**
Education Pak*	**19.95**
Business Pak*	**19.95**

(*Valu-paks have 5 programs)

VALORUM
441 Clyde Ave.
Mtn. View, CA 94043
(415) 968-8500

WITH OUR BASIC COMPILER YOU WON'T BE LEFT WAITING!

FOR C-64
$29.95

SEND CHECK OR MONEY ORDER TO:

JOE LABORATORIES

2116 E. Arapaho, Box #371
Richardson, TX 75081

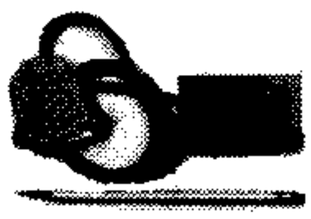
GET THE MOST FROM YOUR VIC-20/C64

CASSETTE INTERFACE

- USE ANY STANDARD CASSETTE RECORDER
- CONNECTS TO THE CASSETTE PORT
- CONTROLS THE CASSETTE MOTOR
- NEEDS NO BATTERIES
- COPY FROM RECORDER TO RECORDER
- THIS IS THE BEST ONE FOR LESS $$$
- ONLY $34.95 POSTPAID ADD $2.50 FOR SHIPPING OUTSIDE US. CANADA. MEXICO

THE MODEM INTERFACE

- CONNECTS TO THE USER I/O PORT
- NEEDS NO BATTERIES/SOLDERING
- CONNECTS ANY STANDARD MODEM
- USE MODEM AUTODIAL/ANSWER FEATURES
- COMES WITH A FREE TYPE IN BASIC TERMINAL PROGRAM
- ONLY $24.95 POSTPAID ADD $2.50 FOR SHIPPING OUTSIDE US. CANADA. MEXICO



CHARGE OR COD ORDERS CALL
1-800-227-3800
ASK FOR OPERATOR 225

BYTESIZE MICRO TECHNOLOGY PO BOX 12309 DEPT DM SEATTLE, WA 98111 (206) 236-BYTE
CALL OR WRITE FOR DEALER INFORMATION

SM-LOS

Loadable Operating Systems from SM are designed to:

maximise the usefulness of expanded memories

provide the optimum environment for memory-greedy programs.

provide an extended vocabulary of BASIC commands.

for cbm 8032 or 8096

TI-99/4A OWNERS

Keyboard Basic
Learn at Home • No Classes

Comprehensive Instruction
Game Programs
Educational Programs
Useful Programs
(Includes Checkbook/Budget)

For Information &
Free Program
CALL TOLL FREE
1-800-241-6083

Basic ***TIPS*** ***AMLIST, Inc.***

4542 Memorial Dr. #202
Atlanta, GA 30032 (404) 292-0576

BONANZA!

COMPUTER TUTOR

Animax presents its effective and proven method of learning computer programming in a simple and easily understood process. Starting with a very small program, you'll learn to expand the program simply, a step at a time, by gradually introducing information.

$14.95

THE ANIMAX VIC-KIT

(for VIC-20 owners)

Includes:

PRINTED MATERIAL:
Instruction manual
Screen Layout sheets
Memory Maps

SOFTWARE:
6 Arcade games
4 Educational games
Custom Character Generation
2 "Make-A-Games"
Plus 12 other information packed programs in easily understandable format.

$24.95

ANIMAX COMPUTER
1111 Las Vegas Blvd. South
Las Vegas, NV 89104
(702) 384-0746

VIC IS A TRADEMARK OF COMMODORE

YOUR NAME HERE

PERSONALIZED COMPUTER PAPER

Printed with your name, club, anything. Paper is white 20# stock and fits all printers using 9½x11 continuous paper [8½x11 when detached]. **500 sheets $14.95, 1000 sheets $24.95. We pay shipping.** Texas orders add 5½% tax. Select ink color: red, blue, brown, gray or canary. Specify name[s] up to 30 letters & spaces. Enclose check or money order. No COD's. Allow 3 weeks. Faster delivery with M/C, Visa phone orders. Write or phone **Personalized Computer Paper C,** Box 20539/San Antonio, Tx. 78220/[512] 227-0585.