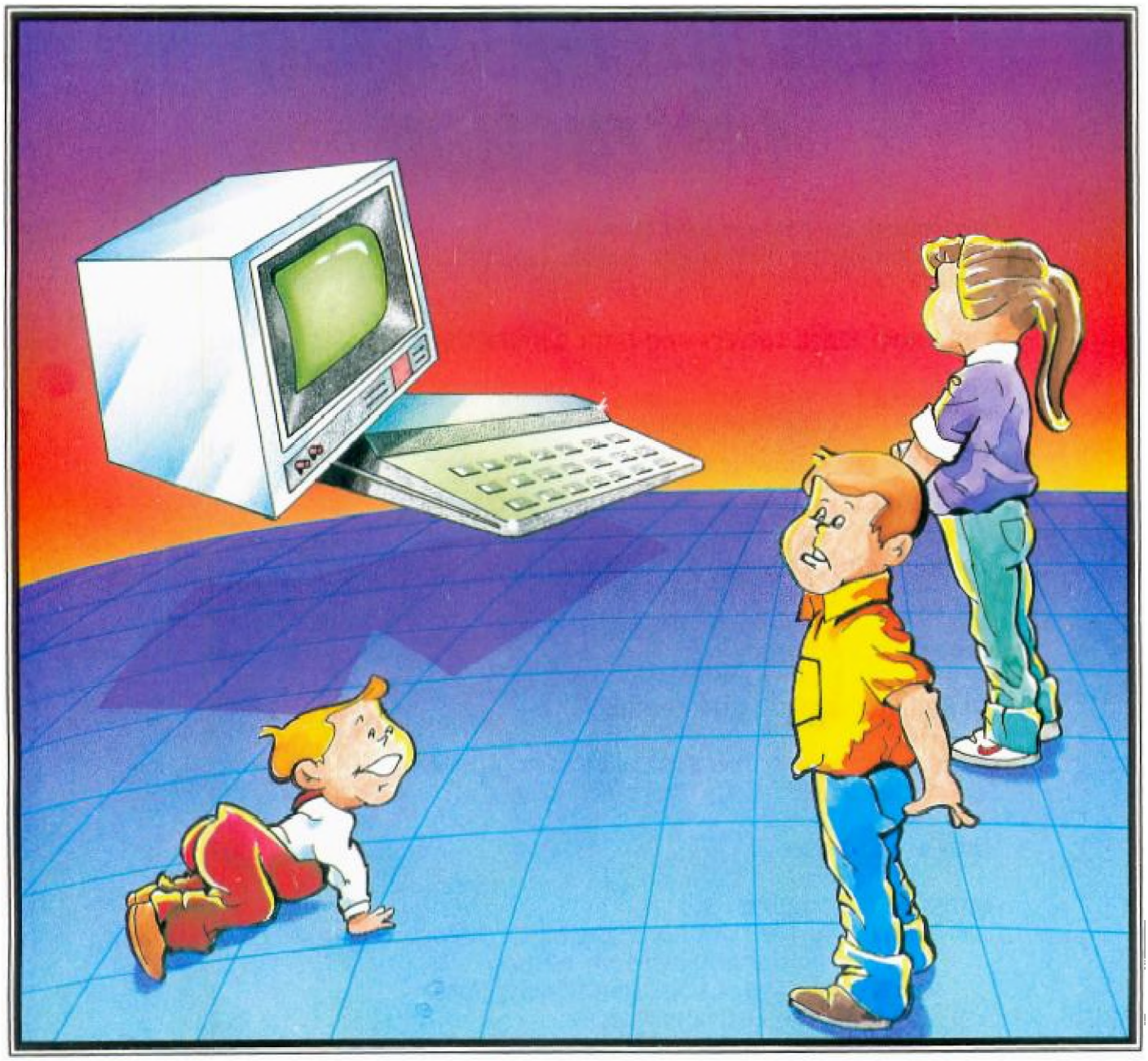
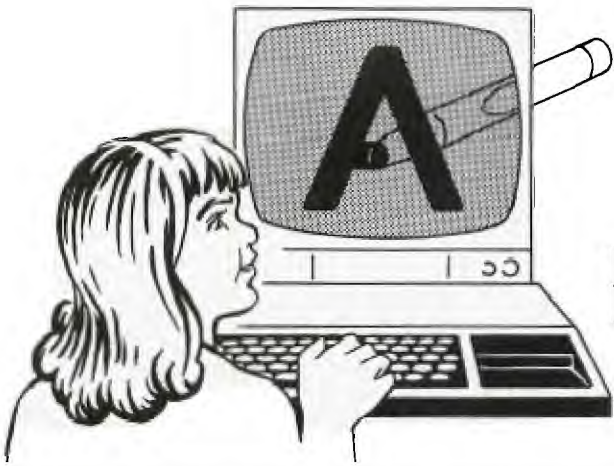
6
Computer-Assisted Instruction



6

Computer-Assisted Instruction

Preschool to college and beyond...
Is the computer a learners' magic wand?



TI BASIC

PRESCHOOL BLOCK LETTERS AND DATA COMPACTION

Most kids aged 100 weeks to 100 years old are fascinated by computers. And small kids are really fascinated by a computer's video screen; it's like a TV, but they can control it. When mine were just learning the alphabet, they would wriggle in between Dad and the computer, then push the "A" key so an "A" would pop onto the screen. But the popping part was the problem. A computer doesn't draw (write) letters on the video screen—it "pops" up the whole letter at once. (Or at least to our slow eyes it "pops" the whole letter at once.) But kids can't just squeeze a crayon and have a letter pop onto a piece of paper. They have to learn a series of hand motions in order to make a recognizable "A" on a piece of paper.

But just maybe the computer could make large letters by popping short line segments in sequence onto the screen if. . . . This was the start of my idea. The finished product is in the program that follows, *Preschool Block Letters.* And the intervening (gory) details are about data compaction.

Most home computers don't have point-addressable graphics, but they do have character graphics that can produce line segments at various angles. Thus, I thought, I would build the letters and numbers from short line segments. Easier said than done. What I really needed to build the letters and numbers was some help. Fortunately, my wife, a teacher, retaught me the correct way to form letters; I, in turn, taught the computer.

Then I had to store about 3500 pieces of information concerned with which line segments go where to form each letter and number. Each piece of information as a number requires 4 to 8 bytes depending on the computer. But *strings* require only one byte for each stored character. And among letters, numbers and punctuation marks, there are enough different characters so that over 40 unique values can be stored using only one byte per value.

Furthermore, strings can be very long, so this helps hold down the overhead to identify each string. Thus, to change the piece of information to the value of a valid ASCII character, I just added a constant to each piece of information. The characters were thus grouped into strings, and the strings stored in DATA statements. The SEG$ and ASC functions retrieve the information as required. And that's how computers came to *draw* rather than *pop* letters.

Note: Make sure ALPHA LOCK is DOWN.

EXPLANATION OF THE PROGRAM
Preschool Block Letters

Lines Nos.	
130-230	Program initialization.
240-330	Scan keyboard looking for a letter or number key.
340-360	Change ASCII code to number between 0 and 35.
360-490	Draw line segments of letter or number in an array.
560-590	Store geometry of "W" in array.
1700-2040	Define line segment characters used to make letters and numbers.
2050-2100	Input word from user.
2110-2320	Have little man hold up letter.
2330-2430	Get key pushed. If it matches letter that man is holding, then draw letter.

```
100 REM  **  PRESCHOOL BLOCK LETTERS  *
    *
110 REM
120 REM
130 CALL SCREEN(8)
140 OPTION BASE 0
150 DIM AR$(35,2)
160 GOSUB 1690
170 GOSUB 490
180 CALL CLEAR
190 INPUT "DO YOU WANT TO WORK WITH
    WHOLE WORDS RATHER THAN      INDIV
    IDUAL LETTERS? (Y/N)":ANS$
200 ANS$=SEG$(ANS$,1,1)
210 IF ANS$="Y" THEN 2040
220 CALL CLEAR
```

```
230 PRINT "PRESS A LETTER OR NUMBER KE
    Y"
240 GOSUB 260
250 GOTO 230
260 CALL KEY(0,KEE,STATUS)
270 IF STATUS=0 THEN 260
280 IF KEE<48 THEN 260
290 IF KEE>90 THEN 260
300 IF -((KEE>=58)*(KEE<=64))THEN 260
310 IF KEE<58 THEN 340
320 IF KEE>64 THEN 330
330 KEE=KEE-7
340 S1=KEE-48
350 CALL CLEAR
360 FOR DELAY=1 TO 10
370 NEXT DELAY
```

```
380 FOR I=1 TO LEN(AR$(S1,1))
390 REM
400 COL=ASC(SEG$(AR$(S1,1),I,1))-40
410 ROW=ASC(SEG$(AR$(S1,2),I,1))-42
420 C=ASC(SEG$(AR$(S1,0),I,1))+63
430 CALL SOUND(100,300,2)
440 CALL HCHAR(ROW,COL,C)
450 FOR DELAY=1 TO 10
460 NEXT DELAY
470 NEXT I
480 RETURN
490 REM
500 REM
510 RESTORE 600
520 FOR S1=0 TO 35
530 READ AR$(S1,1),AR$(S1,2),AR$(S1,0)
540 NEXT S1
550 W1$="WXYWXYWXYWXY"
560 W2$="Z"&CHR$(91)&CHR$(92)
570 W3$=W2$&W2$&W2$&W2$
580 AR$(32,0)=W1$&W3$&W1$&W3$
590 RETURN
600 DATA "321000000000012345677777777765
4"
610 DATA "00123456789:;;;;;:98765432100
"
620 DATA "DECJKSSSSIHBFGEDCKJVVVVHIBGF
"
630 DATA "0000000000000"
640 DATA "0123456789:;"
650 DATA "TTTTTTTTTTTT"
660 DATA "0123456788765432100123456788"
670 DATA "2100000012345678 9:;<<<<<<<<<<"
680 DATA "JCEDRFGBIKCCCCCCCCCRRRRRRRRR"
690 DATA "0123456788765556788876543210
0"
700 DATA "0000000000012345567789:;;;;;;:98
"
710 DATA "PPPPPPPPPCCCCFGBIUKCDEPGFBHI
"
720 DATA "0000001234567896666666666666"
730 DATA "0123455555555550123456789:;"
740 DATA "VVVVVVPPPPPPPPVVVVVMVVVVVV"
750 DATA "000001234567776543210123456 7
"
760 DATA "12345555555678 9:;;;;;;:0000000
"
770 DATA "VVVVVEDRFGBIUKCDEPGFBPPPPPPP
"
780 DATA "4433221100123456788765432 1"
790 DATA "0123456789:;;;;;;:98777777 8"
800 DATA "JKJKJKJKVVBFGPEDCKIGFRRDEC"
810 DATA "0123455544332211 0"
820 DATA "0000001234567 89:;"
830 DATA "PPPPPPJKJKJKJKJKJ"
840 DATA "765432100123456788765432100 1
2345677"
850 DATA "1000001234556667 89:;;;;;;:987
6655432"
860 DATA "BGFRDECJHBFGRFGBIKCDEPGFBHJC
EDEDCKJ"
870 DATA "765432100123456788888888888"
880 DATA "1000001234444443223456789:;"
890 DATA "BGFRDECJHFGPEDCCSSSSSSSSSS"
900 DATA "5544332211006677889 9:;;;;3456
78"
910 DATA "0123456789:;0123456789:;6666
66"
920 DATA "JKJKJKJKJKJKIHIHIHIHIHIHRRRR
RR"
930 DATA "0000000000000012345678876543 21
12345678876543 21"
940 DATA "0123456789:;0000000123455555555
66666789:;;;;;;;;"
950 DATA "VVVVVVVVVVVVRRRRFGBIKCDEPPPP
RRRRFGBIKCDEPPPP"
960 DATA "887654321000000000123456788"
970 DATA "321000000123456789:;;;;;;:98"
980 DATA "HIBGFRDECJKSSSSIHBFGPEDCKJ"
990 DATA "0000000000000012345677888887765
4321"
1000 DATA "0123456789:;0000001234566789:;
;;;;;"
1010 DATA "VVVVVVVVVVVVRRRFGBIHITTKJKCD
EPPPP"
1020 DATA "0000000000000012345671234512 34
567"
1030 DATA "0123456789:;0000000555555;;;;;
;;;"
1040 DATA "VVVVVVVVVVVRRRRRRRPPPPPPPPP
PPP"
1050 DATA "0000000000000012345671 2345"
1060 DATA "0123456789:;0000000055555"
1070 DATA "VVVVVVVVVVVRRRRRRPPPPP"
1080 DATA "887654321000000000123456788888
7"
1090 DATA "321000000123456789:;;;;;;:9877
7"
1100 DATA "HIBGFRDECJKSSSSIHBFGPEDCKJVL
R"
1110 DATA "0000000000000077777777777771234
56"
1120 DATA "0123456789:;0123456789:;5555
55"
1130 DATA "VVVVVVVVVVVVSSSSSSSSSSSSSPPPP
PP"
1140 DATA "00000000000000001 01"
1150 DATA "0123456789:;00;;"
1160 DATA "VVVVVVVVVVVLRMP"
1170 DATA "6666666666654321 00567"
1180 DATA "123456789:;<<<<;:9000"
1190 DATA "UUUUUUUUUUKCDQFBHTPPP"
1200 DATA "0000000000000012345633445566"
1210 DATA "0123456789:;54321045678 9:;"
1220 DATA "VVVVVVVVVVVCCCCCCIHIHIHIH"
1230 DATA "0000000000000012345 67"
1240 DATA "0123456789:;;;;;;;;;;;"
1250 DATA "VVVVVVVVVVVPPPPPPPP"
1260 DATA "0000000000000012348765999999999
9999"
1270 DATA "0123456789:;0123012301230123456 7
89:;"
1280 DATA "VVVVVVVVVVVBBBBCCCCSSSSSSSSS
SSSS"
1290 DATA "0000000000000011223344556677777
77777777"
1300 DATA "0123456789:;0123456789:;0123
456789:;"
1310 DATA "VVVVVVVVVVVIHIHIHIHIHIHSSSSS
SSSSSS"
1320 DATA "43210000000001234567888888888 7
65"
1330 DATA "000123456789:;;;;;;:98765432 1
00"
1340 DATA "RDECJKSSSSIHBFGPEDCKJVVVVHIB
GF"
1350 DATA "0000000000000012345677654321"
1360 DATA "0123456789:;000001234555555"
1370 DATA "VVVVVVVVVVVRRRFGBIKCDEPPP"
1380 DATA "43210000000001234567888888888 7
6578"
1390 DATA "000123456789:;;;;;;:98765432 1
00:;"
1400 DATA "RDECJKSSSSIHBFGPEDCKJVVVVHIB
GFAB"
1410 DATA "0000000000000012345677654321 44
5566"
1420 DATA "0123456789:;000001234555555567
89:;"
1430 DATA "VVVVVVVVVVVRRRFGBIKCDEPPPIH
IHIH"
1440 DATA "8765432100123456788765432 10"
1450 DATA "2100000012345555667 89:;;;;;;:9"
1460 DATA "IBGFRDECJHBFGPFGBIKCDEPGFBH"
1470 DATA "4444444444440123456789"
1480 DATA "0123456789:;0000000000"
1490 DATA "VVVVVVVVVVVRRRRLRRRRR"
1500 DATA "0000000000001234567888888888888"
```

```
1510 DATA "0123456789:;;;;;;:9876543210"
1520 DATA "SSSSSSSSIHBFGPEDCKJVVVVVVVV"
1530 DATA "0011223344555;;;;:99887766"
1540 DATA "0123456789:;0123456789:;"
1550 DATA "IHIHIHIHIHIHJKJKJKJKJKJK"
1560 DATA "000111222333777666555444888899
     9::::;;;;???>>>==<<<"
1570 DATA "0123456789:;0123456789:;0123
     456789:;0123456789:;"
1580 DATA "IHIHIHIHIHIHJKJKJKJKJKJKIHIH
     IHIHIHIHJKJKJKJKJKJK"
1590 DATA "0011223344555554433221100"
1600 DATA "0123456789:;0123456789:;"
1610 DATA "IHIHIHIHIHIHJKJKJKJKJKJK"
1620 DATA "0123498765444444444"
1630 DATA "0123401234567899:;"
1640 DATA "BBBBBCCCCCVVVVVVV"
1650 DATA "0123456789:;:9876543210012 3
     456789:;"
1660 DATA "00000000000000012345678 9:;<====
     ========="
1670 DATA "PPPPPPPPPPPPCCCCCCCCCCCCCCRRR
     RRRRRRRR"
1680 STOP
1690 RESTORE 1750
1700 FOR C=128 TO 156
1710 READ CD$
1720 CALL CHAR(C,CD$)
1730 NEXT C
1740 RETURN
1750 DATA "8142241818244281"
1760 DATA "8040201008040201"
1770 DATA "0102040810204080"
1780 DATA "030C30C"
1790 DATA "000000000030C30C"
1800 DATA "C0300C03"
1810 DATA "00000000C0300C03"
1820 DATA "0808040402020101"
1830 DATA "8080404020201 01"
1840 DATA "0101020204040808"
1850 DATA "1010202040408 08"
1860 DATA "FF0101010101 0101"
1870 DATA "0101010101010101FF"
1880 DATA "8080808080808080FF"
1890 DATA "FF8080808080808 08"
1900 DATA "0000000000000000FF"
1910 DATA "00000000FF"
1920 DATA "FF"
1930 DATA "8080808080808080808"
1940 DATA "1010101010101 01 01"
1950 DATA "0808080808080808 08"
1960 DATA "0101010101010101 01"
1970 DATA "8080804040402020 2"
1980 DATA "2010101008080808 04"
1990 DATA "0404020202020101 01"
2000 DATA "0101010202020204 04"
2010 DATA "0408080808101010 02"
2020 DATA "2020404040808 08"
2030 DATA "82927C10102828 28"
2040 REM LETTER REPEATER
2050 PRINT :"IN THIS ACTIVITY A LITTLE
     MAN WILL HOLD UP EACH LETTEROF A
     WORD YOU INPUT."
2060 PRINT "AFTER HE HAS PICKED UP THE
     LETTER, PUSH THAT KEY ON THEKEYBO
     ARD."
2070 PRINT "YOUR DESIRED WORD MAY BE UP
     TO 16 LETTERS LONG.":
2080 INPUT "INPUT DESIRED WORD: ":ANS$
2090 CALL CLEAR
2100 BL=32
2110 RX=0=5
2120 RX1=7
2130 RX3=8
2140 RXL=6
2150 RY0=21
2160 MAN=156
2170 RYL=20
2180 FOR RX=1 TO LEN(ANS$)
2190 CALL HCHAR(RY0,8+RX,ASC(SEG$(ANS$,
     RX,1)))
2200 NEXT RX
2210 FOR DELAY=1 TO 100
2220 NEXT DELAY
2230 FOR RX=1 TO LEN(ANS$)
2240 CALL GCHAR(RYL,RXL+RX,RC)
2250 CALL HCHAR(6,5,32,448)
2260 CALL VCHAR(RYL,RXL+RX,BL,2)
2270 CALL HCHAR(RY0,RX0+RX,RC)
2280 CALL HCHAR(RY0,RX1+RX,MAN)
2290 CALL GCHAR(RY0,RX3+RX,RC)
2300 CALL HCHAR(RY0,RX3+RX,BL)
2310 CALL HCHAR(RYL,RX1+RX,RC)
2320 IF RC=32 THEN 2410
2330 CALL KEY(0,KEE,STATUS)
2340 IF KEE=RC THEN 2350 ELSE 2330
2350 IF KEE<58 THEN 2370
2360 KEE=KEE-7
2370 S1=KEE-48
2380 GOSUB 380
2390 FOR DELAY=1 TO 200
2400 NEXT DELAY
2410 NEXT RX
2420 GOTO 2080
2430 END
```

HOMEWORK HELPER

5 FRACTIONS

Homework Helper: Students do their class assignments on paper in the usual way. . . . and then can use the Homework Helper to quickly correct their assignments.

The *Homework Helper* series is designed to quickly give answers to students checking their assignments. It is not meant to be a tutorial; it does not teach concepts nor quiz the student. Rather, it gives the answers to the problems without showing all the intermediate steps.

The students are encouraged to do their class assignments on paper in the usual way, writing the problems down and working the problems step-by-step. Then, they can use the *Homework Helper* to correct their assignments quickly.

Fractions

This program, involving fractions, is for correcting the homework problems of elementary school math students (4th, 5th, and 6th graders). Written in TI BASIC, it employs color graphics and sound, and is interactive. There are seven sections, each introduced with a simple color representation of what that section is doing with fractions. Musical phrases from Mendelssohn, Handel, and Beethoven are played at the same time.

1. **Equivalence**. Two fractions are of the form

$$\frac{A}{B} = \frac{C}{D}$$

Any one of the four positions can be the unknown. The user designates the unknown, and inputs the three given values. The computer finds the unknown and prints the equivalent fractions. A student can also use this section to find equivalent ratios.

2. **Simplification**. The user inputs a numerator and a denominator. The computer simplifies (reduces) the fraction or tells if it cannot be simplified.

3. **Multiplication**. The user designates the number of fractions to be multiplied, then enters the numerator and denominator for each one. The computer multiplies them and simplifies the final fraction.

4. **Division**. Two fraction are entered; the first is then divided by the second, and the answer is simplified.

5. **Addition—Like Denominators**. The user specifies the number of fractions to be added, the common denominator and then enters the numerators. The computer adds the numbers and simplifies the result.

6. **Addition—Unlike Denominators**. This section may be used to add fractions with like or unlike denominators. The user specifies the number of fractions up to five (which should be sufficient for elementary school mathematics), and then inputs the numerator and denominator of each. The computer adds the fractions and simplifies the result. A student can also use either Section 5 or 6 for subtraction problems by entering a negative numerator.

7. **Comparisons**. As many as ten fractions may be compared on a number line. The user enters the number of fractions to be compared (up to ten), and then enters the numerator and denominator of each. The computer then arranges the fractions from the smallest to the largest and prints them.

To stop any section of the program press SHIFT C. To restart, enter RUN.

Simplifying Fractions

One basic technique of simplifying fractions is to start with the numerator as the first factor and see if it can be divided evenly into the denominator. If it can, both numerator and denominator are divided by that factor immediately to yield the simplified fraction. If the denominator cannot be evenly divided, the factor is reduced by one, and the numerator and denominator are tested to see if they are divisible by the new factor.

In each successive test, the factor is reduced by one. When both numerator and denominator can finally be evenly divided by the factor, that factor is the greatest common factor. The numerator and denominator are then divided by this factor to yield the reduced fraction

For larger numbers, the technique can take a lot of time. In this program, the algorithm has been made more efficient by first checking to see which is smaller, the numerator or the denominator. In improper fractions the denominator will be smaller. The starting factor, PLIM, is set equal to the smaller number (Statements 1380 to 1410).

Another efficiency technique is not to test all even factors if either numerator or denominator is an odd number. This technique cuts the search time in half. In Statements 1420 to 1450 the step size, S, is set equal to -2 if either the numerator or the denominator is odd; S is set equal to -1 if both numerator and denominator are even numbers.

The simplifying algorithm is implemented with a FOR-NEXT loop. The starting trial factor is reduced by the step size, S, to a lower limit of 2 in line 1460.

Within the loop, Statements 1460 to 1510 set A = NS/P (where NS is the numerator) and set B = DS/P (where DS is the denominator). Then they check to see if A = INT(A); if equal, then B = INT(B) is checked. If both statements are true, the simplified fraction is A/B. Otherwise, P is incremented by S, and the loop continues. If the lower limit is reached without finding a successful factor, the user is notified that the fraction cannot be simplified (Statements 1520-1540).

When combining several fractions in multiplication or addition, another efficiency technique sets the starting

factor equal to the largest denominator of the orginal fractions (Statements 2250-2340). The common denominator may be much larger than the original denominators, but the largest factor will always be the largest original denominator.

Comparisons

The schoolroom technique for comparing fractions is to find the common denominator and then compare the adjusted numerators. This technique is far too slow for computers, especially when comparing many fractions and/or fractions with large numbers. A very fast technique which achieves the same result is to compute and compare the decimal equivalents of the fractions.

As the fractions are read in, the numerator NNN (I) is divided by the denominator DD (I) and stored as a decimal fraction in two identical arrays, FRC (I) and FRD (I) (Statements 5170-5230). A standard sort routine sorts the first array FRC from the smallest to the largest. The subscripts are changed as the decimal fractions are arranged in order (Statements 5250-5330).

The first element of FRC is compared with each element of the second array, FRD. When a match is made, the subscript value J is used to retrieve the numerator and denominator of the corresponding fraction for printing. The process is repeated in order for each element in the FRC array (Statements 5340-5390).

EXPLANATION OF THE PROGRAM

Homework Helper: Fractions

Line Nos.

160-170 Sets T and T2 for the time in the music statements.
180-250 Defines characters and colors in four different character sets for use in graphics.
260-390 Prints title screen,"HOMEWORK HELPER".
410-550 Prints "Fractions" and blinks an outline of asterisks around it.
580-640 Prints the menu screen for the seven sections of the program.
670-730 The user presses a key to choose which of the 7 sections is wanted, and the computer branches to that section.
749-890 Prints the screen for Equivalence.
900-960 Asks for the unknown, A,B,C, or D.
970-1190 Depending on which is the unknown, asks for the given values and calculates the unknown. If the unknown is not a whole number, it will be rounded to two decimal places.
1200-1230 Prints the equivalent fractions.
1240-1310 Asks if there is another problem or to stop. If "2" is pressed, the menu screen is returned.
1320-1350 Prints screen for Simplifying.
1360-1370 Asks for the fraction.
1460-1540 Simplifies and prints the result.
1550-1620 Continue, or go to menu screen.
1530-1650 Prints screen for Multiplying.
1660-1710 Asks for the fractions.
1720-1770 Multiplies the fractions.
1780-1800 Prints the problem and the simplified answer.
1810-1880 Continue, or go to menu screen.
1890-1930 Subroutine for printing the problem.
1940-2150 Subroutine for simplifying and printing.
2160-2360 Subroutine for sorting and simplifying. These three subroutines are used for simplifying and printing in other sections of the program also.
2370-2340 Prints screen for Dividing.
2440-2470 Asks for the two fractions.
2480-2490 Performs division.
2500-2540 Prints problem and simplified solution.
2550-2620 Continue, or go to menu screen.
2630-2680 Prints screen for Adding with like denominators.
2690-2760 Asks for fractions and adds the numerators.
2770-2820 Prints the problem and the simplified sum.
2830-2990 Continue, or go to menu screen.
2910-2960 Prints screen for Adding with unlike denominators.
2970-3090 Asks for the fractions and calculates a common denominator.
3100-3150 Adds the adjusted numerators and prints the problem and the simplified result.
3160-3230 Continue, or go to menu screen.
3240-4090 Sound subroutines musical phrases.
4100-5020 Draws color graphics for each title screen.
5030-5120 Prints screen for Comparisons.
5130-5230 Asks for fractions and converts fractions to decimals.
5240-5330 Sorts fractions from the smallest to the largest.
5340-5390 Prints fractions in order.
5400-5470 Continue, or go to menu screen.
5480-5670 Music and graphics for Comparisons.

To stop the program, press SHIFT C (BREAK). For the student's convenience, at the end of each problem he can choose to do another problem of the same type or go to the menu screen and do a problem of a different type.

```
100 REM **********************
110 REM *                    *
120 REM *  HOMEWORK HELPERS: *
130 REM *       FRACTIONS    *
140 REM *                    *
150 REM **********************
160 REM
170 REM
180 REM
190 T=300
200 T2=150
210 CALL CHAR(128,"F")
220 CALL COLOR(13,7,7)
230 CALL CHAR(136,"F")
240 CALL COLOR(14,11,11)
250 CALL CHAR(144,"F")
260 CALL COLOR(15,5,5)
270 CALL CHAR(152,"F")
280 CALL COLOR(16,13,13)
290 CALL CLEAR
300 CALL COLOR(2,16,7)
310 DATA 72,79,77,69,87,79,82,75
320 RESTORE 310
330 FOR Y=9 TO 23 STEP 2
```

```
340 READ L
350 CALL HCHAR(7,Y,L)
360 NEXT Y
370 DATA 72,69,76,80,69,82
380 RESTORE 370
390 FOR Y=11 TO 21 STEP 2
400 READ L
410 CALL HCHAR(10,Y,L)
420 NEXT Y
430 GOSUB 3270
440 CALL HCHAR(14,10,42,13)
450 CALL VCHAR(15,22,42,3)
460 CALL VCHAR(15,10,42,3)
470 CALL HCHAR(18,10,42,13)
480 DATA 70,82,65,67,84,73,79,78,83
490 RESTORE 480
500 FOR Y=12 TO 20
510 READ L
520 CALL HCHAR(16,Y,L)
530 NEXT Y
540 FOR I=1 TO 30
550 CALL COLOR(2,7,16)
560 CALL COLOR(2,16,7)
570 NEXT I
```

```
580 GOSUB 3380
590 CALL COLOR(2,2,1)
600 CALL CLEAR
610 PRINT "1 EQUIVALENCE"
620 PRINT "2 SIMPLIFY"
630 PRINT "3 MULTIPLY"
640 PRINT "4 DIVIDE"
650 PRINT "5 ADD-LIKE DENOMINATORS"
660 PRINT "6 ADD-UNLIKE DENOMINATORS"
670 PRINT "7 COMPARISONS":::::
680 GOSUB 4890
690 GOSUB 3470
700 CALL KEY(0,CHOICE,STATUS)
710 IF STATUS<=0 THEN 700
720 IF CHOICE<49 THEN 700
730 IF CHOICE>55 THEN 700
740 CHOICE=CHOICE-48
750 CALL CLEAR
760 ON CHOICE GOTO 770,1350,1660,2400,
    2660,2940,5060
770 DATA 69,81,85,73,86,65,76,69,78,67
    ,69
780 RESTORE 770
790 CALL CLEAR
800 FOR Y=11 TO 21
810 READ L
820 CALL HCHAR(9,Y,L)
830 NEXT Y
840 CALL CHAR(96,"00000000FF")
850 GOSUB 4130
860 DATA 14,13,65,15,13,96,16,13,66,15
    ,16,61,14,19,67,15,19,96,16,19,68
870 RESTORE 860
880 FOR I=1 TO 7
890 READ X,Y,L
900 CALL HCHAR(X,Y,L)
910 NEXT I
920 GOSUB 3550
930 INPUT "WHICH IS THE UNKNOWN?":X$
940 IF X$="A" THEN 1000
950 IF X$="B" THEN 1060
960 IF X$="C" THEN 1120
970 IF X$="D" THEN 1180
980 PRINT "CHOSE A,B,C,OR D"
990 GOTO 930
1000 INPUT "ENTER B=":B
1010 INPUT "ENTER C=":C
1020 INPUT "ENTER D=":D
1030 A=B*C/D+.005
1040 A=1E-2*(INT(A*1E2))
1050 GOTO 1230
1060 INPUT "ENTER A=":A
1070 INPUT "ENTER C=":C
1080 INPUT "ENTER D=":D
1090 B=A*D/C+.005
1100 B=1E-2*(INT(B*1E2))
1110 GOTO 1230
1120 INPUT "ENTER A=":A
1130 INPUT "ENTER B=":B
1140 INPUT "ENTER D=":D
1150 C=A*D/B+.005
1160 C=1E-2*(INT(C*1E2))
1170 GOTO 1230
1180 INPUT "ENTER A=":A
1190 INPUT "ENTER B=":B
1200 INPUT "ENTER C=":C
1210 D=B*C/A+.005
1220 D=1E-2*(INT(D*1E2))
1230 CALL CLEAR
1240 PRINT "     ";A;C
1250 PRINT "     =  ------":::
1260 PRINT "          ";B;D:::
1270 PRINT "PRESS 1 FOR NEXT PROBLEM"
1280 PRINT "PRESS 2 TO STOP"
1290 CALL KEY(0,K,STATUS)
1300 IF STATUS<=0 THEN 1290
1310 IF K<>49 THEN 1330
1320 CALL CLEAR
1325 GOTO 780
1330 IF K=50 THEN 590
1340 GOTO 1290
1350 PRINT "** SIMPLIFYING FRACTIONS **
     "
1360 GOSUB 4230
1370 PRINT :::
1380 GOSUB 3650
1390 INPUT "NUMERATOR?":NS
1400 INPUT "DENOMINATOR?":DS
1410 IF DS>NS THEN 1440
1420 PLIM=DS
1430 GOTO 1450
1440 PLIM=NS
1450 S=-2
1460 IF DS/2=INT(DS/2)THEN 1480
1470 IF NS/2=INT(NS/2)THEN 1490
1480 S=-1
1490 FOR P=PLIM TO 2 STEP S
1500 A=NS/P
1510 IF A<>INT(A)THEN 1540
1520 B=DS/P
1530 IF B=INT(B)THEN 1570
1540 NEXT P
1550 PRINT :NS;"/";DS;" CANNOT BE SIMPL
     IFIED":::::
1560 GOTO 1580
1570 PRINT :NS;"/";DS;" = ";A;"/";B:::
     :::
1580 PRINT "PRESS 1 FOR NEXT PROBLEM"
1590 PRINT "PRESS 2 TO STOP"
1600 CALL KEY(0,K,STATUS)
1610 IF STATUS<=0 THEN 1600
1620 IF K<>49 THEN 1640
1630 CALL CLEAR
1635 GOTO 1350
1640 IF K=50 THEN 590
1650 GOTO 1600
1660 PRINT "** MULTIPLYING FRACTIONS **
     ":::::
1670 GOSUB 4430
1680 GOSUB 3750
1690 INPUT "HOW MANY FRACTIONS? ":F
1700 FOR I=1 TO F
1710 PRINT "FRACTION";I
1720 INPUT "     NUMERATOR = ":NN(I)
1730 INPUT "     DENOMINATOR = ":DD(I)
1740 NEXT I
1750 TN=1
1760 TD=1
1770 FOR I=1 TO F
1780 TN=TN*NN(I)
1790 TD=TD*DD(I)
1800 NEXT I
1810 PRINT :::"** MULTIPLY **":::
1820 GOSUB 1920
1830 GOSUB 2190
1840 PRINT :::"PRESS 1 FOR NEXT PROBLEM
     "
1850 PRINT "PRESS 2 TO STOP"
1860 CALL KEY(0,K,STATUS)
1870 IF STATUS<=0 THEN 1860
1880 IF K<>49 THEN 1900
1890 CALL CLEAR
1895 GOTO 1660
1900 IF K=50 THEN 590
1910 GOTO 1860
1920 FOR I=1 TO F
1930 PRINT NN(I);"/";DD(I)
1940 NEXT I
1950 PRINT "-----------------"
1960 RETURN
1970 IF TN<TD THEN 2000
1980 PLIM=TD
1990 GOTO 2010
2000 PLIM=TN
2010 FOR P=PLIM TO 2 STEP -1
2020 A=TN/P
2030 IF A<>INT(A)THEN 2060
2040 B=TD/P
```

```
2050 IF B=INT(B)THEN 2090
2060 NEXT P
2070 A=TN
2080 B=TD
2090 IF A>=B THEN 2120
2100 PRINT : :A;"/";B: : : :
2110 GOTO 2160
2120 IF B=1 THEN 2170
2130 C=INT(A/B)
2140 R=A-C*B
2150 PRINT C;" ";R;"/";B: : : :
2160 RETURN
2170 PRINT A: : : :
2180 RETURN
2190 FOR I=1 TO F
2200 P=DD(I)
2210 A=TN/P
2220 IF A<>INT(A)THEN 2270
2230 B=TD/P
2240 IF B<>INT(B)THEN 2270
2250 TN=A
2260 TD=B
2270 NEXT I
2280 SW=0
2290 FOR I=1 TO F-1
2300 IF DD(I)<=DD(I+1)THEN 2350
2310 J=DD(I)
2320 DD(I)=DD(I+1)
2330 DD(I+1)=J
2340 SW1=1
2350 NEXT I
2360 IF SW=1 THEN 2280
2370 PLIM=DD(F)
2380 GOSUB 2010
2390 RETURN
2400 PRINT "** DIVIDING FRACTIONS **":
: :
2410 GOSUB 4520
2420 PRINT "THE FIRST FRACTION IS"
2430 PRINT "DIVIDED BY THE "
2440 PRINT "SECOND FRACTION": :
2450 PRINT "(N1/D1) / (N2/D2)": :
2460 GOSUB 3830
2470 INPUT "ENTER N1 = ":N1
2480 INPUT "ENTER D1 = ":D1
2490 INPUT "ENTER N2 = ":N2
2500 INPUT "ENTER D2 = ":D2
2510 TN=N1*D2
2520 TD=D1*N2
2530 PRINT : : :N1;"/";D1
2540 PRINT "_____":
2550 PRINT N2;"/";D2: : :
2560 PRINT "EQUALS": :
2570 GOSUB 1970
2580 PRINT :"PRESS 1 FOR NEXT PROBLEM"
2590 PRINT "PRESS 2 TO STOP"
2600 CALL KEY(0,K,STATUS)
2610 IF STATUS<=0 THEN 2600
2620 IF K<>49 THEN 2640
2630 CALL CLEAR
2635 GOTO 2400
2640 IF K=50 THEN 590
2650 GOTO 2600
2660 PRINT "** ADDING FRACTIONS **": :
2670 GOSUB 4590
2680 PRINT "THIS SECTION ADDS"
2690 PRINT "FRACTIONS THAT ALL HAVE"
2700 PRINT "THE SAME DENOMINATOR.": : :
2710 GOSUB 3910
2720 INPUT "HOW MANY FRACTIONS? ":F
2730 INPUT "WHAT IS THE DENOMINATOR?":T
D
2740 PRINT "ENTER THE NUMERATORS"
2750 TN=0
2760 FOR I=1 TO F
2770 INPUT NN(I)
2780 TN=TN+NN(I)
2790 NEXT I
2800 FOR I=1 TO F
2810 DD(I)=TD
2820 NEXT I
2830 PRINT : : :
2840 GOSUB 1920
2850 GOSUB 1970
2860 PRINT "PRESS 1 FOR NEXT PROBLEM"
2870 PRINT "PRESS 2 TO STOP"
2880 CALL KEY(0,K,STATUS)
2890 IF STATUS<=0 THEN 2880
2900 IF K<>49 THEN 2920
2910 CALL CLEAR
2915 GOTO 2660
2920 IF K=50 THEN 590
2930 GOTO 2880
2940 PRINT "** ADDING FRACTIONS **"
2950 GOSUB 4740
2960 PRINT : : :"THIS SECTION ADDS"
2970 PRINT "FRACTIONS THAT MAY HAVE"
2980 PRINT "UNLIKE DENOMINATORS": : :
2990 GOSUB 4020
3000 INPUT "HOW MANY FRACTIONS? ":F
3010 IF F<=5 THEN 3040
3020 PRINT "SORRY, I CAN ONLY HANDLE UP
TO 5"
3030 GOTO 3000
3040 TN=0
3050 TD=1
3060 FOR I=1 TO F
3070 PRINT "FRACTION";I
3080 INPUT " NUMERATOR = ":NN(I)
3090 INPUT " DENOMINATOR = ":DD(I)
3100 TD=TD*DD(I)
3110 NEXT I
3120 PRINT : :
3130 FOR I=1 TO F
3140 C=TD/DD(I)
3150 TN=TN+NN(I)*C
3160 NEXT I
3170 GOSUB 1920
3180 GOSUB 2190
3190 PRINT "PRESS 1 FOR NEXT PROBLEM"
3200 PRINT "PRESS 2 TO STOP"
3210 CALL KEY(0,K,STATUS)
3220 IF STATUS<=0 THEN 3210
3230 IF K<>49 THEN 3250
3240 CALL CLEAR
3245 GOTO 2940
3250 IF K=50 THEN 590
3260 GOTO 3210
3270 CALL SOUND(T,880,2,698,5)
3280 CALL SOUND(T,932,2,784,5)
3290 CALL SOUND(T,784,2,659,5)
3300 CALL SOUND(T,880,2,698,5)
3310 CALL SOUND(T,698,2,587,5)
3320 CALL SOUND(T2,784,2)
3330 CALL SOUND(T2,698,2)
3340 CALL SOUND(T2,659,2)
3350 CALL SOUND(T2,784,2)
3360 CALL SOUND(T,698,2,587,5)
3370 RETURN
3380 CALL SOUND(T,440,2)
3390 CALL SOUND(T2,466,2)
3400 CALL SOUND(T2,523,2)
3410 CALL SOUND(T2,587,2)
3420 CALL SOUND(T2,523,2)
3430 CALL SOUND(T,466,2)
3440 CALL SOUND(T,440,2)
3450 CALL SOUND(1000,392,2,330,5)
3460 RETURN
3470 CALL SOUND(T2,440,2)
3480 CALL SOUND(T2,466,2)
3490 CALL SOUND(T,523,2)
3500 CALL SOUND(T,440,2)
3510 CALL SOUND(T,587,2)
3520 CALL SOUND(T,784,2)
3530 CALL SOUND(500,659,2)
3540 RETURN
3550 CALL SOUND(T,698,2)
3560 CALL SOUND(T,932,2)
```

```
3570 CALL SOUND(T,784,2)
3580 CALL SOUND(T,880,2)
3590 CALL SOUND(T2,932,2)
3600 CALL SOUND(T2,880,2)
3610 CALL SOUND(T2,784,2)
3620 CALL SOUND(T2,880,2)
3630 CALL SOUND(500,698,2)
3640 RETURN
3650 CALL SOUND(T2,659,2)
3660 CALL SOUND(T2,587,2)
3670 CALL SOUND(T,523,2)
3680 CALL SOUND(T,440,2)
3690 CALL SOUND(T,698,2,440,5)
3700 CALL SOUND(T,784,2,587,5)
3710 CALL SOUND(T2,698,2,392,5)
3720 CALL SOUND(T2,659,2)
3730 CALL SOUND(1000,698,2,440,5)
3740 RETURN
3750 DATA 262,349,392,440,523,440,392,3
     49,392
3760 RESTORE 3750
3770 FOR I=1 TO 9
3780 READ M
3790 CALL SOUND(T2,M,2)
3800 NEXT I
3810 CALL SOUND(500,440,2)
3820 RETURN
3830 CALL SOUND(600,262,10)
3840 CALL SOUND(600,311,7)
3850 CALL SOUND(450,392,4)
3860 CALL SOUND(150,349,4)
3870 CALL SOUND(300,311,6)
3880 CALL SOUND(300,294,8)
3890 CALL SOUND(500,262,10)
3900 RETURN
3910 CALL SOUND(T,523,2)
3920 CALL SOUND(T,440,2)
3930 CALL SOUND(T2,440,2)
3940 CALL SOUND(T2,494,2)
3950 CALL SOUND(T,523,2)
3960 CALL SOUND(T2,494,2)
3970 CALL SOUND(T2,523,2)
3980 CALL SOUND(T2,494,2)
3990 CALL SOUND(T2,392,2)
4000 CALL SOUND(1000,440,2)
4010 RETURN
4020 CALL SOUND(400,440,8)
4030 CALL SOUND(200,392,8)
4040 CALL SOUND(200,440,7)
4050 CALL SOUND(400,587,6)
4060 CALL SOUND(200,523,5)
4070 CALL SOUND(200,587,4)
4080 CALL SOUND(400,494,3)
4090 CALL SOUND(200,440,4)
4100 CALL SOUND(200,494,5)
4110 CALL SOUND(500,392,6)
4120 RETURN
4130 CALL HCHAR(12,4,128,3)
4140 CALL HCHAR(13,4,128,3)
4150 FOR Y=4 TO 6
4160 CALL VCHAR(14,Y,144,4)
4170 NEXT Y
4180 CALL VCHAR(11,27,136,3)
4190 CALL VCHAR(11,28,136,3)
4200 CALL VCHAR(14,27,152,6)
4210 CALL VCHAR(14,28,152,6)
4220 RETURN
4230 FOR X=10 TO 14 STEP 2
4240 FOR Y=9 TO 13 STEP 2
4250 CALL HCHAR(X,Y,144)
4260 CALL HCHAR(X,Y-1,136)
4270 NEXT Y
4280 NEXT X
4290 FOR X=11 TO 13 STEP 2
4300 FOR Y=8 TO 12 STEP 2
4310 CALL HCHAR(X,Y,144)
4320 CALL HCHAR(X,Y+1,136)
4330 NEXT Y
4340 NEXT X
```

```
4350 CALL HCHAR(12,16,61)
4360 FOR Y=19 TO 21
4370 CALL VCHAR(10,Y,136,5)
4380 NEXT Y
4390 FOR Y=22 TO 24
4400 CALL VCHAR(10,Y,144,5)
4410 NEXT Y
4420 RETURN
4430 Y=6
4440 FOR I=1 TO 5
4450 CALL VCHAR(10,Y,136,2)
4460 CALL VCHAR(10,Y+1,136,2)
4470 CALL VCHAR(12,Y,128,4)
4480 CALL VCHAR(12,Y+1,128,4)
4490 Y=Y+5
4500 NEXT I
4510 RETURN
4520 CALL HCHAR(10,11,136,13)
4530 CALL HCHAR(11,11,136,13)
4540 FOR X=12 TO 14
4550 CALL HCHAR(X,11,144,13)
4560 NEXT X
4570 CALL VCHAR(7,17,93,11)
4580 RETURN
4590 CALL HCHAR(10,8,128,2)
4600 CALL VCHAR(11,8,152,4)
4610 CALL VCHAR(11,9,152,4)
4620 CALL VCHAR(10,13,128,4)
4630 CALL VCHAR(10,14,128,4)
4640 CALL HCHAR(14,13,152,2)
4650 CALL VCHAR(10,18,128,2)
4660 CALL VCHAR(10,19,128,2)
4670 CALL VCHAR(12,18,152,3)
4680 CALL VCHAR(12,19,152,3)
4690 CALL VCHAR(10,23,128,3)
4700 CALL VCHAR(10,24,128,3)
4710 CALL VCHAR(13,23,152,2)
4720 CALL VCHAR(13,24,152,2)
4730 RETURN
4740 CALL VCHAR(10,8,128,4)
4750 CALL VCHAR(14,8,136,3)
4760 CALL VCHAR(10,12,144,2)
4770 CALL VCHAR(10,13,144,2)
4780 CALL VCHAR(12,12,128,3)
4790 CALL VCHAR(12,13,128,3)
4800 CALL HCHAR(10,17,136,3)
4810 CALL HCHAR(11,17,152,3)
4820 CALL HCHAR(12,17,152,3)
4830 CALL HCHAR(13,17,152,3)
4840 CALL VCHAR(10,23,152,2)
4850 CALL VCHAR(10,24,152,2)
4860 CALL VCHAR(12,23,144,4)
4870 CALL VCHAR(12,24,144,4)
4880 RETURN
4890 CALL HCHAR(4,15,128,2)
4900 CALL HCHAR(5,14,128,3)
4910 CALL HCHAR(6,13,128,4)
4920 CALL HCHAR(7,13,128,4)
4930 CALL HCHAR(4,17,136,2)
4940 CALL HCHAR(5,17,136,3)
4950 CALL HCHAR(6,17,136,4)
4960 CALL HCHAR(7,17,136,4)
4970 CALL HCHAR(8,17,152,4)
4980 CALL HCHAR(9,17,152,4)
4990 CALL HCHAR(10,17,152,3)
5000 CALL HCHAR(11,17,152,2)
5010 CALL HCHAR(11,15,144,2)
5020 CALL HCHAR(10,14,144,3)
5030 CALL HCHAR(9,13,144,4)
5040 CALL HCHAR(8,13,144,4)
5050 RETURN
5060 DATA 67,79,77,80,65,82,73,83,79,78
     ,83,32
5070 RESTORE 5060
5080 FOR Y=11 TO 22
5090 READ L
5100 CALL HCHAR(14,Y,L)
5110 NEXT Y
5120 GOSUB 5510
```

```
5130 DIM NNN(10),DDD(10),FRC(10),FRD(10
     )
5140 PRINT "THIS ARRANGES FRACTIONS"
5150 PRINT "FROM SMALLEST TO LARGEST.":
     :
5160 INPUT "HOW MANY FRACTIONS?":NF
5170 IF NF<11 THEN 5200
5180 PRINT "SORRY; UP TO 10 ONLY."
5190 GOTO 5160
5200 FOR I=1 TO NF
5210 PRINT :: :"FRACTION ";I
5220 INPUT "      NUMERATOR:":NNN(I)
5230 INPUT "    DENOMINATOR:":DDD(I)
5240 FRC(I)=NNN(I)/DDD(I)
5250 FRD(I)=FRC(I)
5260 NEXT I
5270 PRINT : : : :
5280 SW=0
5290 FOR I=1 TO NF-1
5300 IF FRC(I)<=FRC(I+1)THEN 5350
5310 FF=FRC(I)
5320 FRC(I)=FRC(I+1)
5330 FRC(I+1)=FF
5340 SW=1
5350 NEXT I
5360 IF SW=1 THEN 5280
5370 FOR I=1 TO NF
5380 FOR J=1 TO NF
5390 IF FRC(I)=FRD(J)THEN 5410
5400 NEXT J
5410 PRINT I;"  ";NNN(J);"/";DDD(J)
```

```
5420 NEXT I
5430 PRINT : : :"PRESS 1 FOR NEXT PROBL
     EM"
5440 PRINT "PRESS 2 TO STOP"
5450 CALL KEY(0,K,STATUS)
5460 IF STATUS<=0 THEN 5450
5470 IF K<>49 THEN 5490
5480 CALL CLEAR
5485 GOTO 5070
5490 IF K=50 THEN 590
5500 GOTO 5450
5510 CALL SOUND(400,330,2,262,5)
5520 CALL VCHAR(4,8,136,3)
5530 CALL VCHAR(4,9,136,3)
5540 CALL SOUND(100,330,2)
5550 CALL HCHAR(7,8,144,2)
5560 CALL SOUND(100,262,3)
5570 CALL SOUND(400,330,1)
5580 CALL VCHAR(4,15,128,4)
5590 CALL VCHAR(8,15,152,2)
5600 CALL SOUND(100,330,2)
5610 CALL SOUND(100,262,3)
5620 CALL SOUND(400,330,1)
5630 CALL VCHAR(4,23,152,3)
5640 CALL VCHAR(4,24,152,3)
5650 CALL SOUND(200,392,5)
5660 CALL VCHAR(7,23,136,3)
5670 CALL SOUND(200,524,3)
5680 CALL VCHAR(7,24,136,3)
5690 CALL SOUND(400,660,1)
5700 RETURN
```

HOMEWORK HELPER

DIVISION

Homework Helper: Students do their class assignments on paper in the usual way. . . . and then use the Homework Helper to quickly correct their assignments.

Division gives the answers to three types of homework problems an elementary school student may encounter: division with a remainder, division with a decimal in the quotient, and division to convert a fraction to a decimal.

Only the answers are given, not the step-by-step process of long division. The student is encouraged to do the homework—writing each step in the division process and then using this program to check the answers. Music and graphics enhance the interaction.

1. Division with Remainder. Most math problems can simply be corrected with a calculator. However if there is a remainder, a calculator converts it to a decimal equivalent. This program keeps the answer in quotient-plus-remainder form. The student enters the divisor and dividend; the quotient and remainder are printed.

2. Division with Decimal. Usually after students master the idea of a remainder, they are taught how to place a decimal and keep dividing. In this section, a student enters the divisor and dividend; the quotient with a decimal fraction is printed.

3. Convert Fraction to Decimal. A fraction is converted to a decimal by dividing the numerator by the denominator. The student enters the numerator then the denominator; the equivalent decimal fraction is returned.

After each problem, a student may enter another problem of the same type. If there are no more problems of the same kind or the student wishes to stop, he enters zero and the menu screen will return.

EXPLANATION OF THE PROGRAM
Homework Helper: Division

Line Nos.	
130-770	Prints title screen and blinks color while special graphics characters are defined.
780-1680	Plays music; prints menu screen and branches appropriately for student's response.
1690-1810	Subroutine to print labels of division problem.
1820-1940	Routine for division with remainder.
1950-2050	Routine for division with decimal.
2060-2320	Routine for converting fraction to decimal.

```
100 REM  *  HOMEWORK HELPER  *
110 REM  *       DIVISION     *
120 REM
130 T=300
140 CALL CLEAR
150 CALL CHAR(96,"0000078444444444478")
```

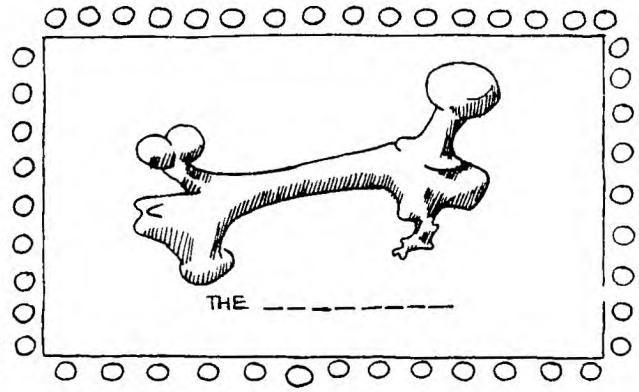
```
160 CALL CHAR(97,"0000038101010101038")
170 CALL CHAR(98,"0000044444444428100")
180 PRINT TAB(7);"H O M E W O R K"
190 CALL CHAR(99,"0000038443008443800")
200 PRINT :TAB(9);"H E L P E R"
210 CALL CHAR(100,"0000C444444444C")
220 CALL CHAR(101,"0000784444784844")
```

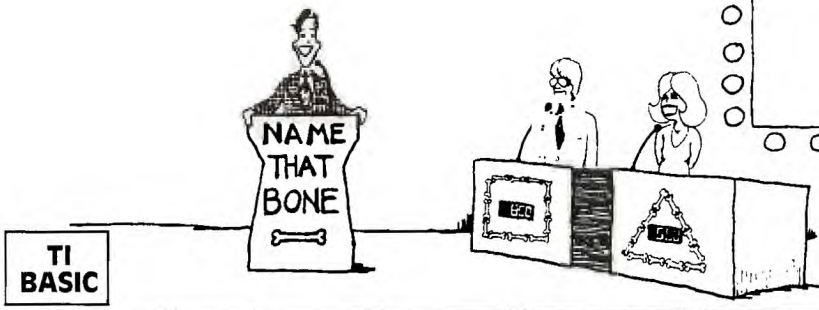
```
 230 PRINT ::::::::::TAB(7));"D I V I S I O
     N":::::::::
 240 CALL CHAR(102,"3F1010008081010102")
 250 CALL CHAR(103,"FF007844444444478")
 260 CALL COLOR(2,7,16)
 270 CALL HCHAR(14,7,42,19)
 280 CALL VCHAR(15,7,42,3)
 290 CALL VCHAR(15,25,42,3)
 300 CALL HCHAR(18,7,42,19)
 310 CALL COLOR(2,16,7)
 320 CALL CHAR(104,"FF0038101010101038")
 330 CALL CHAR(105,"FF00444444444281")
 340 CALL COLOR(2,7,16)
 350 CALL CHAR(106,"FF007C4078404078C")
 360 CALL CHAR(107,"FF00446454544C44")
 370 CALL COLOR(2,16,7)
 380 CALL CHAR(108,"0172FCFEFEFE7E3C")
 390 CALL CHAR(112,"38444454C3C")
 400 CALL COLOR(2,7,16)
 410 CALL CHAR(113,"4444444444438")
 420 CALL CHAR(114,"7C444444447C")
 430 CALL COLOR(2,16,7)
 440 CALL CHAR(115,"7C10101010101")
 450 CALL CHAR(116,"381010101038")
 460 CALL COLOR(2,7,16)
 470 CALL CHAR(117,"7C4078404078C")
 480 CALL CHAR(118,"446454544C44")
 490 CALL COLOR(2,16,7)
 500 CALL CHAR(119,"0078444447848844")
 510 CALL CHAR(109,"00000040C1C3870E")
 520 CALL COLOR(2,7,16)
 530 CALL CHAR(110,"0103070E1C387")
 540 CALL CHAR(111,"80C0E07038 1C0E07")
 550 CALL COLOR(2,16,7)
 560 CALL CHAR(136,"80C0E0F0F8FCFEFF")
 570 CALL CHAR(137,"7F3F1F0F07030301")
 580 CALL COLOR(2,7,16)
 590 CALL CHAR(121,"000000000003070F0F")
 600 CALL COLOR(2,16,7)
 610 CALL CHAR(122,"030F3FFFFFFFFFFF")
 620 CALL CHAR(123,"C0F0FCFFFFFFFFFF")
 630 CALL COLOR(2,7,16)
 640 CALL CHAR(124,"000000000C0E0F0F")
 650 CALL CHAR(125,"1F1F3F3F7F7FFFFF")
 660 CALL COLOR(2,16,7)
 670 CALL CHAR(126,"F8F8FCFCFEFEFFFF")
 680 CALL CHAR(127,"FFFF7F7F3F3F1F1F")
 690 CALL COLOR(2,7,16)
 700 CALL CHAR(128,"FFFFFEFEFCFCF8F8")
 710 CALL CHAR(129,"0F0F07 03")
 720 CALL COLOR(2,16,7)
 730 CALL CHAR(130,"FFFFFFFFFF3F0F03")
 740 CALL CHAR(131,"FFFFFFFFFFFCF0C")
 750 CALL COLOR(2,7,16)
 760 CALL CHAR(132,"F0F0E0C")
 770 CALL COLOR(2,2,1)
 780 CALL CLEAR
 790 CALL CHAR(120,"FFFFFFFFFFFFFFFF")
 800 CALL COLOR(12,1,1)
 810 CALL COLOR(13,1,1)
 820 CALL SOUND(T/2,262,5)
 830 CALL VCHAR(15,6,120,5)
 840 CALL SOUND(T/2,294,5)
 850 CALL HCHAR(15,4,121)
 860 CALL SOUND(T,330,4,131,10)
 870 CALL HCHAR(15,5,122)
 880 CALL HCHAR(15,7,123)
 890 CALL HCHAR(15,8,124)
 900 CALL SOUND(T,330,4,196,10)
 910 CALL HCHAR(16,4,125)
 920 CALL HCHAR(16,5,120,3)
 930 CALL HCHAR(16,8,126)
 940 CALL SOUND(T/2,330,4)
 950 CALL HCHAR(17,4,120,5)
 960 CALL SOUND(T/2,392,4)
 970 CALL HCHAR(18,4,127)
 980 CALL SOUND(T,349,3,147,9)
 990 CALL HCHAR(18,5,120,3)
1000 CALL HCHAR(18,8,128)
1010 CALL HCHAR(19,4,129)
1020 CALL HCHAR(19,5,130)
1030 CALL SOUND(T,349,3,196,9)
1040 CALL HCHAR(19,7,131)
1050 CALL HCHAR(19,8,132)
1060 CALL COLOR(12,7,1)
1070 CALL COLOR(13,7,1)
1080 CALL SOUND(T/2,349,3)
1090 CALL COLOR(14,15,1)
1100 CALL COLOR(9,5,1)
1110 CALL SOUND(T/2,330,3)
1120 CALL HCHAR(13,9,136)
1130 CALL HCHAR(14,9,137)
1140 CALL SOUND(T,294,3,175,9)
1150 CALL HCHAR(14,10,136)
1160 CALL HCHAR(15,10,137)
1170 CALL HCHAR(15,11,136)
1180 CALL HCHAR(16,11,137)
1190 CALL SOUND(T/2,294,3,196,9)
1200 CALL HCHAR(16,12,136)
1210 CALL HCHAR(17,12,137)
1220 CALL SOUND(T/2,330,3)
1230 CALL HCHAR(17,13,136)
1240 CALL SOUND(T/2,349,2)
1250 CALL HCHAR(18,13,137)
1260 CALL SOUND(T/2,370,2)
1270 CALL HCHAR(18,14,136)
1280 CALL SOUND(T,392,1,165,7)
1290 CALL COLOR(10,7,1)
1300 CALL HCHAR(18,15,109)
1310 CALL HCHAR(19,14,110)
1320 CALL HCHAR(19,15,111)
1330 CALL SOUND(T,392,1,196,7)
1340 CALL HCHAR(20,16,111)
1350 CALL COLOR(11,5,1)
1360 CALL SOUND(T/2,440,2)
1370 CALL SOUND(T/2,494,2)
1380 CALL SOUND(T,523,1,165,7)
1390 CALL SOUND(T,523,1,196,7)
1400 CALL SOUND(T/2,587,2)
1410 CALL CHAR(120,"7FBFDFEFF7FBFDFE")
1420 CALL SOUND(T/2,523,2)
1430 CALL SOUND(T,494,1,175,7)
1440 CALL SOUND(T,784,1,196,7)
1450 CALL HCHAR(17,18,61)
1460 CALL SOUND(T/2,587,1)
1470 CALL SOUND(T/2,523,1)
1480 CALL SOUND(T,494,0,175,6)
1490 CALL HCHAR(13,22,108,7)
1500 CALL SOUND(T,880,0,196,6)
1510 CALL HCHAR(15,22,108,7)
1520 CALL HCHAR(17,22,108,7)
1530 CALL SOUND(T,494,0)
1540 CALL HCHAR(19,22,108,7)
1550 CALL HCHAR(21,22,108,4)
1560 CALL SOUND(T*2,523,1,165,7,131,9)
1570 CALL HCHAR(24,1,67)
1580 CALL HCHAR(24,2,72)
1590 PRINT "OOSE:"
1600 PRINT :"1 DIVISION WITH REMAINDER"
1610 PRINT :"2 DIVISION WITH DECIMAL"
1620 PRINT :"3 CONVERT FRACTION TO DECI
     MA"
1630 CALL HCHAR(23,31,76)
1640 PRINT :"4 END PROGRAM"
1650 CALL KEY(0,K,S)
1660 IF (K<49)+(K>52)THEN 1650
1670 CALL CLEAR
1680 ON K-48 GOTO 1820,1950,2060,2320
1690 DATA 96,97,98,97,99,100,101,102,10
     3,104,105,104,103,106,107,103,32
1700 DATA 112,113,114,115,116,117,118,1
     15,32
1710 CALL COLOR(10,5,1)
1720 RESTORE 1690
1730 FOR I=7 TO 23
1740 READ G
1750 CALL HCHAR(19,I,G)
1760 NEXT I
```

```
1770 FOR I=15 TO 23
1780 READ G
1790 CALL HCHAR(18,I,G)
1800 NEXT I
1810 RETURN
1820 PRINT "DIVISION WITH REMAINDER":::
     :::::::
1830 GOSUB 1710
1840 CALL HCHAR(18,25,119)
1850 CALL HCHAR(18,26,35,2)
1860 INPUT "DIVISOR:  ":D
1870 IF D=0 THEN 780
1880 INPUT "DIVIDEND:  ":N
1890 C=INT(N/D)
1900 R=N-C*D
1910 PRINT :"QUOTIENT =";C;"  R";R
1920 PRINT :::"ENTER NEXT PROBLEM"
1930 PRINT "OR '0' TO STOP.":::::::::::
1940 GOTO 1830
1950 PRINT "DIVISION WITH DECIMAL":::::
     :::::
1960 GOSUB 1710
1970 CALL HCHAR(18,23,46)
1980 CALL HCHAR(18,24,35,3)
1990 INPUT "DIVISOR:  ":D
2000 IF D=0 THEN 780
2010 INPUT "DIVIDEND:  ":N
2020 PRINT :"QUOTIENT =";N/D
2030 PRINT :::"ENTER NEXT PROBLEM"
2040 PRINT "OR '0' TO STOP.":::::::::::
```

```
2050 GOTO 1960
2060 PRINT "CONVERT FRACTION TO DECIMAL
     ":::::::::::
2070 CALL CHAR(120,"FFFFFFFFFFFFFFFF")
2080 CALL CHAR(138,"000000FFFF")
2090 CALL HCHAR(16,8,120,4)
2100 CALL HCHAR(17,8,120,4)
2110 CALL HCHAR(18,7,138,6)
2120 CALL HCHAR(19,8,120,4)
2130 CALL HCHAR(20,8,120,4)
2140 CALL HCHAR(21,8,120,4)
2150 CALL HCHAR(18,15,61)
2160 CALL HCHAR(18,18,46)
2170 CALL COLOR(10,13,13)
2180 CALL HCHAR(18,20,108)
2190 CALL HCHAR(18,22,108)
2200 CALL HCHAR(18,24,108)
2210 PRINT :::
2220 INPUT "NUMERATOR:  ":N
2230 IF N=0 THEN 780
2240 INPUT "DENOMINATOR:  ":D
2250 IF D<>0 THEN 2280
2260 PRINT "SORRY, CANNOT DIVIDE BY ZER
     O":::
2270 GOTO 2240
2280 PRINT :N;"/";D;"=";N/D
2290 PRINT :::"ENTER NEXT PROBLEM"
2300 PRINT "OR ENTER '0' TO STOP."
2310 GOTO 2210
2320 END
```

NAME THAT BONE





Time to review Ezekiel's "Dry Bones" song: "Leg bone connected to the hip bone. . ." Or was it the ankle bone? Or what bone is where?? This program is designed to teach the names of the major bones of the human body and where they are located, and then turn what could be a dry, repetitious drill into an enjoyable game of *Name That Bone.*

The menu screen of the program gives the choice of major parts of the body, head, arms, torso, and legs, or end the program. Each section will label the main bones of the part of the body chosen:

1. HEAD: frontal, parietal, zygomatic, temporal, maxilla, mandible.
2. ARMS: humerus, ulna, radius, carpus, metacarpus, phalanges.
3. TORSO: spine, ribs, clavicle, scapula, sternum, ilium, ischium, sacrum, coccyx.
4. LEGS: femur, tibia, fibula, patella, tarsus, metatarsus, phalanges.

You may study the labeled diagram of the bones as long as you wish, then press ENTER. The labels will be erased and it will be your turn to *Name That Bone.* The bones are listed in a random order at the left of the screen for your choice of answers. A bone will be chosen randomly and will blink red and white until you press a number corresponding to the name of the bone. If you are correct, an arpeggio is played; if you are incorrect, a noise is sounded. You must press the correct answer to continue, and it won't take long for you to learn the names of your bones.

After each bone is chosen once, you will be asked TRY AGAIN? (Y/N). If the response is N, the program returns to the menu screen. If the response is Y, the names of the bones will be rearranged and the bones will be chosen in a different order.

Programming Techniques

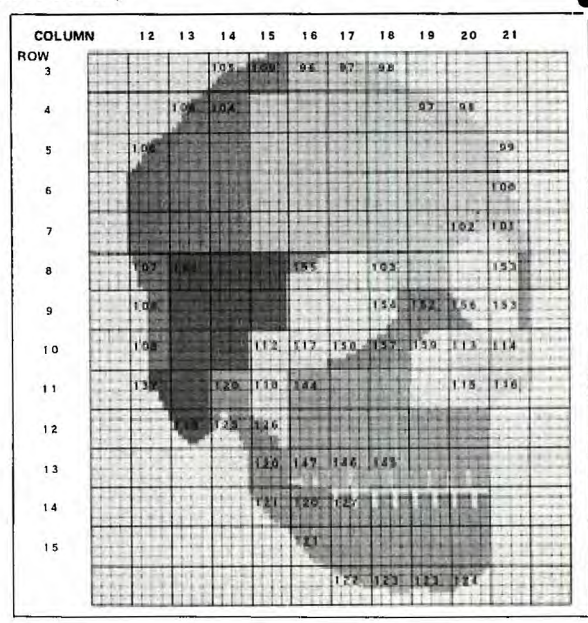
There are four main parts of the body from which to choose, and each part uses the same program logic, so subroutines are used. The subroutines are located at the beginning program. For some microcomputers, execution is faster for subroutines called closer to the beginning; however, the speed in TI BASIC does not seem to depend upon the location of the subroutine.

For each part of the body, different characters are defined. The appropriate DATA statement is RESTOREd, then the subroutine to define characters (lines 160-210) is called. After the labels for the bones are printed, the bones are drawn, again RESTOREing the corresponding DATA statement and calling a subroutine (lines 320-360).

The main procedure is in Lines 370-980. The program will read from DATA the names of the bones and the character set number, then randomly print the bones and choose the bones for the quiz.

The graphics characters were designed so that a specific bone could be blinked by using CALL COLOR statements. The characters of one bone must be in one character set, and another bone in another character set. When the main part of the body is first drawn, all the characters are yellow, but as the bone is chosen, the characters in that set will blink. An example is shown with the skull bones.

(NOTE: The wrist and hand bones are known either as the *carpus* and *metacarpus* or *carpals* and *metacarpals*. The carpals are the elements of the carpus (wrist bone). You may wish to relabel these parts to be consistent with the way you teach them.)



EXPLANATION OF THE PROGRAM

Name That Bone

Line Nos.

150 Branches to title screen.

160-210 Subroutine reads C and C$ from DATA to define graphics characters.

220-310 Subroutine prints PRESS ENTER and waits for the user to respond.

320-360 Subroutine reads DATA to draw graphics.

370-980 Subroutine for main program logic.

370-390 For the number of bones R, reads the name of the bone and the corresponding character set number.

400-520 Randomly prints the names of the bones for the multiple-choice answers and arranges the corresponding character set number and answer number.

530-580 Prints NAME THAT BONE at the top of the screen.

590-660 Randomly chooses a bone and blinks it red and white while waiting for the user to press the answer.

670-780 If the answer is correct, plays an arpeggio and goes to the next bone; if the answer is incorrect, sounds a noise and awaits another key press.

790-980 Prints TRY AGAIN? (Y/N) and branches appropriately after Y or N is pressed.

990-1100 Defines graphic characters for head.

1110-1120 Labels head bones.

1130-1260 Draws skull and waits for user to press ENTER.

1270-1300 Clears labels.

1310-1350 Main procedure for head.

1360-1440 Defines character for arm.

1450-1510 Labels arm bones.

1520-1590 Draws arm bones and waits for user to press enter.

1600-1630 Clears labels.

1640-1680 Main procedure for arm.

1690-1830 Defines characters and colors for torso.

1840-1860 Labels torso bones.

1870-2030 Draws torso bones and waits for user to press ENTER.

2040-2090 Clears labels.

2100-2170 Main procedure for torso.

2180-2250 Defines characters for leg.

2260-2270 Labels leg bones.

2280-2380 Draws leg bones and waits for user to press ENTER.

2390-2420 Clears labels.

2430-2470 Main prodecure for leg.

2500-2610 Prints title screen and draws stick figure.

2620-2710 First time through the program defines the first character in each character set as a solid block. It then asks if instructions are desired.

2720-2800 Prints instructions and waits for user to press ENTER.

2810-2900 Prints choices of head, arms, torso, legs, or end program.

2910-3000 Waits for user's choice and branches appropriately.

```
100 REM  ********************
110 REM  *  NAME  THAT  BONE  *
120 REM  ********************
130 REM
140 REM
150 GOTO  2490
160 FOR  I=1  TO  N
170 READ  C,C$
180 CALL  CHAR(C,C$)
190 NEXT  I
200 CALL  CLEAR
210 RETURN
220 DATA  80,82,69,83,83,32,60,69,78,84
    ,69,82,62,32
230 RESTORE  220
240 FOR  Y=19  TO  32
250 READ  G
260 CALL  HCHAR(24,Y,G)
270 NEXT  Y
280 CALL  KEY(0,K,S)
290 IF  K<>13  THEN  280
300 CALL  HCHAR(24,19,32,13)
310 RETURN
320 FOR  I=1  TO  N
330 READ  X,Y,G,R
340 CALL  HCHAR(X,Y,G,R)
350 NEXT  I
360 RETURN
370 FOR  I=1  TO  R
380 READ  BONE$(I),B(I)
390 NEXT  I
400 RANDOMIZE
410 FOR  I=1  TO  R
420 RR=INT(RND*R+1)
430 IF  BONE$(RR)=""  THEN  420
440 B$(RR)=BONE$(RR)
450 BB(RR)=B(RR)
460 ANS(RR)=I
470 CALL  HCHAR(15+I,2,48+I)
480 FOR  J=1  TO  LEN(B$(RR))
490 CALL  HCHAR(15+I,J+3,ASC(SEG$(B$(RR
    ),J,1)))
500 NEXT  I
510 BONE$(RR)=""
520 NEXT  I
```

```
530 DATA  78,65,77,69,32,84,72,65,84,32
    ,66,79,78,69,32
540 RESTORE  530
550 FOR  Y=9  TO  23
560 READ  G
570 CALL  HCHAR(1,Y,G)
580 NEXT  Y
590 FOR  I=1  TO  R
600 RR=INT(RND*R+1)
610 IF  B$(RR)=""  THEN  600
620 CALL  HCHAR(14,2,63,3)
630 CALL  KEY(0,K,S)
640 CALL  COLOR(BB(RR),16,1)
650 CALL  COLOR(BB(RR),7,1)
660 IF  S<1  THEN  630
670 IF  K-48=ANS(RR)THEN  700
680 CALL  SOUND(500,-5,1)
690 GOTO  630
700 CALL  HCHAR(14,2,32,3)
710 CALL  SOUND(150,262,1)
720 CALL  SOUND(150,330,1)
730 CALL  SOUND(150,392,1)
740 CALL  SOUND(150,330,1)
750 CALL  SOUND(200,262,1)
760 CALL  COLOR(BB(RR),12,1)
770 B$(RR)=""
780 NEXT  I
790 DATA  84,82,89,32,65,71,65,73,78,63
    ,32
800 RESTORE  790
810 FOR  Y=22  TO  32
820 READ  G
830 CALL  HCHAR(23,Y,G)
840 NEXT  Y
850 CALL  HCHAR(24,26,40)
860 CALL  HCHAR(24,27,89)
870 CALL  HCHAR(24,28,47)
880 CALL  HCHAR(24,29,78)
890 CALL  HCHAR(24,30,41)
900 CALL  KEY(0,K,S)
910 IF  K=78  THEN  2490
920 IF  K<>89  THEN  900
930 FOR  Y=16  TO  24
940 CALL  HCHAR(Y,2,32,12)
950 NEXT  Y
```

```
 960 CALL HCHAR(23,22,32,10)
 970 CALL HCHAR(24,26,32,5)
 980 RETURN
 990 RESTORE 1000
1000 DATA 97,F0FCFFFFFFFFFFFF,98,0000C0
     E0F8FCFEFF,99,0080C0C0E0E0E0E,100,
     E0E0F0F0F8F8FCFC
1010 DATA 101,FCFCFEFE3F1F0F07,102,FFFF
     FFFFFFEF8F0C,103,FFFF7F1F1F0F0707,1
     09,0F7FFFFFFFFFFFFF
1020 DATA 105,0000000031F3F7FFF,106,0107
     0F0F1F1F3F7F,107,7F7F3F3F3F3F3F1F,
     108,0F0F0F0F0F0F0F0F
1030 DATA 137,0F0F0F0F0F030301010,138,FFFF
     7F7F3F1E0C,155,FFFEFCF,153,03030303
     030303030303,159,F0C0808
1040 DATA 154,03030303030303070F,156,0080
     C0C0E0E0F0F8,157,0F3FFFFFFFFFFFFE,
     158,0000000010300FFFFF
1050 DATA 113,7C7E7F7F3F3F1F1F,114,0F0F
     FFFFFFFFFFFF,115,1F1F0F0F0F0F1F1F,
     116,FFFFFFFFFEE0C0C
1060 DATA 117,0000000080E0FCFFFF,118,FFFF
     FFFFF3F1F07,121,FF7F3F1F1F0F0701,
     122,7F1F0701
1070 DATA 123,FFFFFFFF0F,124,FEFCF0C08,
     125,CFCF8787030301010,126,0080E0F0F
     0F8F8F8
1080 DATA 127,00BBBBBBFFFFFFFFF,145,FFFF
     FFF7777777777,146,FFFF7B7B7B7B7B,147,
     FFFFFFFFFFFDB1BC
1090 N=37
1100 GOSUB 160
1110 PRINT "  PARIETAL":  :TAB(20);"FRONT
     AL":  :  :"  TEMPORAL"
1120 PRINT  TAB(20);"ZYGOMATIC":  :  :TAB(
     19);"MAXILLA":  :  :  :"  MANDIBLE"
     :  :  :  :  :  :  :
1130 RESTORE 1140
1140 DATA 3,14,105,1,3,15,109,1,3,16,96
     ,1,3,17,97,1,3,18,98,1,4,13,106,1,
     4,14,104,1
1150 DATA 4,15,96,4,4,19,97,1,4,20,98,1
     ,5,12,106,1,5,13,104,2,5,15,96,6,5
     ,21,99,1
1160 DATA 6,12,104,3,6,15,96,6,6,21,100
     ,1,7,12,104,3,7,15,96,5,7,20,102,1
     ,7,21,101,1
1170 DATA 8,12,107,1,8,13,136,3,8,16,15
     5,1,8,18,103,1,8,19,96,1,8,21,153,
     1,9,12,108,1
1180 DATA 9,13,136,3,9,18,154,1,9,19,15
     2,1,9,20,156,1,9,21,153,1,10,12,10
     8,1,10,13,136,2
1190 DATA 10,15,112,1,10,16,117,1,10,17
     ,158,1,10,18,157,1,10,19,159,1,10,
     20,113,1,10,21,114,1
1200 DATA 11,12,137,1,11,13,136,1,11,14
     ,120,1,11,15,118,1,11,16,144,3,11,
     20,115,1,11,21,116,1
1210 DATA 12,13,138,1,12,14,125,1,12,15
     ,126,1,12,16,144,5,13,15,120,1,13,
     16,147,1,13,17,146,1
1220 DATA 13,18,145,3,14,15,121,1,14,16
     ,120,1,14,17,127,4,15,16,121,1,15,
     17,120,4
1230 DATA 16,17,122,1,16,18,123,2,16,20
     ,124,1,16,21,32,1
1240 N=66
1250 GOSUB 320
1260 GOSUB 230
1270 RESTORE 1280
1280 DATA 4,3,32,8,6,22,32,7,9,4,32,8,1
     0,22,32,9,13,21,32,7,16,9,32,8
1290 N=6
1300 GOSUB 320
1310 R=6
1320 DATA FRONTAL,9,PARIETAL,10,ZYGOMAT
     IC,11,MANDIBLE,12,TEMPORAL,14,MAXI
     LLA,15
1330 RESTORE 1320
1340 GOSUB 370
1350 GOTO 1330
1360 RESTORE 1370
1370 DATA 97,0001030303070707070707,98,0F0F0F
     0F0F0F0F0F,99,0707070703030301,100,8
     1C3C7EFFFFFFFFF
1380 DATA 101,F7F3F0F0F0F0F3F7FFF,102,FFFF
     FFFFF7F7E3C1,113,0000000000000FFFFFFF,
     121,7E7E7E7E007E7E7E
1390 DATA 129,FEFEFEFEFEFE00000000,130,FEFE
     FEFEFE0000FE,131,FEFEFEFE0000FEFEFEFE,
     137,FFFFFFFFE000080C
1400 DATA 138,FFFFFFFFFF0000FF,139,80C0
     C0C080000000FE,140,FFFFFFFFF0000FFFFF,
     141,FFFFFF0000FFFFFF
1410 DATA 142,FFFFFF0000C0E0E0E0,143,C0C0
     8,145,0000001070F1F7FFF,146,1878F8F
     8F0E0C08
1420 DATA 64,0000001F1010101,94,1010101
     0101010101,95,1010101011F
1430 N=23
1440 GOSUB 160
1450 PRINT TAB(21);"@PHALANG":TAB(12);"
     RADIUS  ^"
1460 CALL HCHAR(22,31,69)
1470 CALL HCHAR(22,32,83)
1480 PRINT  :  :  :TAB(12);"ULNA  ^_METACA
     RPU"
1490 CALL HCHAR(23,31,83)
1500 PRINT  "  HUMERUS":  :TAB(18);"_CARPU
     S":  :  :  :  :  :  :  :  :  :
1510 CALL VCHAR(12,20,94,2)
1520 RESTORE 1530
1530 DATA 8,2,97,1,9,2,98,1,10,2,99,1,8
     ,3,96,8,9,3,96,8,10,3,96,8,8,11,10
     0,1
1540 DATA 9,11,101,1,10,11,102,1,8,12,1
     04,8,9,12,113,8,10,12,112,8,8,20,1
     21,1,9,20,121,1
1550 DATA 10,20,121,1,7,21,145,1,8,21,1
     29,1,9,21,130,1,10,21,131,1,7,22,1
     46,1,8,22,138,2
1560 DATA 9,22,140,2,8,24,139,1,9,24,13
     7,1,10,22,141,1,10,23,142,1,10,24,
     143,1,10,29,32,1
1570 N=27
1580 GOSUB 320
1590 GOSUB 230
1600 N=7
1610 RESTORE 1620
1620 DATA 6,23,32,10,7,14,32,6,7,23,32,
     1,11,14,32,18,12,4,32,19,13,20,32,
     1,14,20,32,7
1630 GOSUB 320
1640 R=6
1650 DATA HUMERUS,9,RADIUS,10,ULNA,11,C
     ARPUS,12,METACARPUS,13,PHLANGES,14
1660 RESTORE 1650
1670 GOSUB 370
1680 GOTO 1660
1690 RESTORE 1700
1700 DATA 43,FFFF00FFFFFFFFFFF,33,FFFFFF
     FF1F1F1F1F0F,34,FFFFFFFFFE0F8FCFE,35,
     FFFFFFFFF
1710 DATA 36,FFFFFFFF071F3F7F,37,FFFFFF
     FFF8F8F8F8F8,97,0707070707070707,100
     ,E0E0E0E0E0E0E0E0
1720 DATA 98,00C0F0FCFEFFFFFF,101,000030
     F3F7FFFFFFFF,99,FF1F0701,102,FFF8E0
     8,105,3FFFFFFF1F1FFFFF
1730 DATA 106,FFF7E7F7F7F7F7FF,107,FCFF
     FFFF8F8FFFF,108,0707070FFFFFF0101,
     109,E0E0E0FFFFFF8080
```

```
1740 DATA 113,000400FFFFFF,114,000FF3F03
     C0C0FFFF,117,C0F07BC00303FFFF,115,
     0000C0118FFF
1750 DATA 116,00000CF0F0F,119,6000301F1
     F0F,127,0001030707F1F3F7F,125,0000C
     0E0F0F3FCFE
1760 DATA 123,7F3F1F0F070301,127,FEFDFB
     F0E0C08,133,EFF7E7E7C381,132,F7EFE
     7E7C381,105,7DBC3E3E3C08
1770 DATA 122,000000FFFFFFFFFF,137,7C7E
     3F3F0F0F0701,141,3E7EFCFCF0F0E08,1
     38,0001010107FFFFFF
1780 DATA 140,0000000E3FFFFFF,158,FFFF
     FFF7F7E3C1,120,FFC1DDF9F3E7CFC1,15
     1,FFFFFFC1FDFDE1FD
1790 DATA 150,00C1E1F7F7F7F7,134,0083
     87EFEFEFEFEF,131,EFEFEFEFEFEFEFEF,
     135,F7F7F7F7F7F7F7F7
1800 N=42
1810 GOSUB 150
1820 CALL COLOR(1,12,1)
1830 CALL COLOR(2,12,1)
1840 PRINT TAB(8);"CLAVICLE";TAB(2
     2);"SCAPULA";TAB(31);"STERNUM"
1850 PRINT TAB(22);"RIBS";TAB(
     16);"SPINE";TAB(2);"SACRUM"
1860 PRINT TAB(20);"ILIUM";TAB(5);"COCCYX";
     TAB(18);"ISCHIUM"
1870 RESTORE 1990
1880 CALL VCHAR(2,17,65,10)
1890 DATA 8,11,35,1,8,12,36,1,8,13,35,3
     ,8,16,36,1,8,18,36,1,8,19,35,3,8,2
     2,36,1,8,23,37,1
1900 DATA 5,11,97,1,5,12,96,1,5,15,98,1
     ,5,15,114,1,5,16,105,1,5,17,105,1,
     5,18,107,1,5,19,117,1
1910 DATA 5,21,101,1,5,22,96,1,5,23,100
     ,1,8,11,97,1,8,12,96,2,8,14,113,2,
     8,19,100,1
1920 DATA 8,17,104,1,5,18,100,1,5,19,11
     3,2,5,21,96,2,8,25,100,1,7,12,99,1
     ,7,13,113,9
1930 DATA 7,17,104,1,7,22,102,1,8,12,11
     3,11,8,17,104,1,8,12,115,4,8,16,11
     6,1,9,18,119,1,9,19,115,4
1940 DATA 10,11,113,5,10,18,115,5,11,11
     ,113,4,11,15,116,1,11,19,119,1,11,
     20,113,4,12,10,115,5
1950 DATA 12,20,113,5,13,10,113,4,13,14
     ,116,1,13,20,119,1,13,21,113,4,16,
     13,121,1,16,14,120,3
1960 DATA 16,16,130,1,16,17,120,1,16,18
     ,134,1,16,19,120,2,16,21,125,1,17,
     13,120,5,17,16,131,1
1970 DATA 17,17,120,1,17,18,135,1,17,19
     ,120,3,18,13,120,3,18,16,133,1,18,
     17,151,1,18,18,132,1
1980 DATA 18,19,120,5,19,13,125,1,19,14
     ,120,2,19,17,145,1,19,19,120,2,19,
     21,127,1,20,14,123,1
1990 DATA 20,15,120,1,20,16,122,3,20,19
     ,120,1
2000 DATA 20,20,127,1,21,15,137,1,21,16
     ,138,1,21,17,139,1,21,18,140,1,21,
     19,141,1,24,30,32,1
2010 N=84
2020 GOSUB 320
2030 GOSUB 230
2040 RESTORE 2050
2050 DATA 3,20,32,8,7,3,32,9,8,24,32,7,
     10,24,32,4,14,10,32,5,10,22,32,5,1
     7,3,32,8
2060 DATA 10,3,32,8,21,20,32,7,5,17,104
     ,1,17,17,128,1,18,17,128,1,18,17,1
     28,1
2070 N=13
2080 GOSUB 320
```

```
2090 CALL CHAR(145,"7F7F3E3E1C08")
2100 DATA CLAVICLE,1,SCAPULA,8
     STERNUM,10,RIBS,11,ILIUM,12,SACRUM
     ,15,COCCYX,16,ISCHIUM,14
2110 N=9
2120 RESTORE 2100
2130 GOSUB 370
2140 CALL VCHAR(16,15,121)
2150 CALL VCHAR(17,12,120,2)
2160 CALL VCHAR(19,13,123)
2170 GOTO 2120
2180 RESTORE 2190
2190 DATA 97,F0F8F8F8F0E0C08,98,071F7F7
     F7F7F7F7F7F,99,3F1F0F0F0707030301,100,
     FFFFFFFFFFFFFFC
2200 DATA 105,7E7F7F7F7F7F7F3E,113,7F7F
     FFFF7F7F7F7F,114,00C0FFFFFFFFFFFF,
     115,FFFFFFFFFFFFFE0
2210 DATA 122,FFCF8F0F0F0F0F0F,121,0F0F
     0F0F0F0F0F,129,3E7FFFFFFFFFFF7F,
     137,1C3F1F0F0F0703030F0F
2220 DATA 138,5B5B5A5F0F0F5F5FFF,139,FFFF
     8F0F0F0F0F,140,FFFF0FFFF0FFFFFF,14
     5,7CFF0F05F5F5E07E
2230 DATA 146,F00FF073,147,1F0F03,130,0
     0F0FEFFFFFFFFFF
2240 N=18
2250 GOSUB 160
2260 PRINT TAB(14);"FEMUR";
     TAB(14);"PATELLA";TAB(7);"TIBI
     A";
2270 PRINT TAB(14);"FIBULA";TAB
     (14);"TARSUS";TAB(9);"METATARSUS
     PEALANGES"
2280 CALL VCHAR(2,14,96,2)
2290 CALL VCHAR(2,15,96,8)
2300 CALL VCHAR(13,14,112,7)
2310 CALL VCHAR(14,15,121,7)
2320 RESTORE 2330
2330 DATA 2,15,98,1,2,16,97,1,5,13,99,1
     ,10,15,100,1,11,14,100,1,11,15,105
     ,1,12,14,113,1
2340 DATA 12,15,114,1,13,15,122,1,20,14
     ,115,1,21,14,120,1,21,15,130,1,22,
     14,137,1,22,15,138,1
2350 DATA 23,15,130,1,23,16,140,2,23,18
     ,105,1,24,18,146,1,24,17,147,1,24,
     30,32,1
2360 N=20
2370 GOSUB 320
2380 GOSUB 230
2390 RESTORE 2400
2400 DATA 4,16,32,3,11,16,32,7,14,6,32,
     5,16,16,32,6,21,16,32,6,25,5,32,10
     ,23,19,32,9
2410 N=7
2420 GOSUB 320
2430 R=7
2440 DATA FEMUR,8,PATELLA,10,TIBIA,11,F
     IBULA,12,TARSUS,13,METATARSUS,14,P
     HALANGES,15
2450 RESTORE 2440
2460 GOSUB 370
2470 GOTO 2450
2480 STOP
2490 CALL CLEAR
2500 PRINT TAB(7);"NAME THAT BONE"
2510 A$="FFFFFFFFFFFFFFFF"
2520 CALL CHAR(96,A$)
2530 CALL COLOR(9,7,7)
2540 CALL HCHAR(7,15,96,3)
2550 CALL HCHAR(8,15,96,3)
2560 CALL HCHAR(9,15,96,3)
2570 CALL VCHAR(10,15,96,6)
2580 CALL HCHAR(11,13,96,7)
2590 CALL VCHAR(15,15,96,6)
```

```
2600 CALL VCHAR(15,17,96,6)
2610 CALL COLOR(2,2,1)
2620 IF FLAG=2 THEN 2830
2630 FOR I=1 TO 7
2640 CALL COLOR(9+I,12,1)
2650 CALL CHAR(96+8*I,A$)
2660 NEXT I
2670 FLAG=2
2680 PRINT "INSTRUCTIONS? (Y/N)"
2690 CALL KEY(0,K,S)
2700 IF K=78 THEN 2830
2710 IF K<>89 THEN 2690
2720 CALL CLEAR
2730 PRINT "YOU MAY STUDY THE NAMES OF"
     : :"THE BONES AS LONG AS YOU"
2740 PRINT :"WISH. THEN PRESS <ENTER>."
2750 PRINT : :"THE LABELS WILL CLEAR, T
     HEN": :"IT IS YOUR TURN TO NAME"
2760 PRINT :"THAT BONE - CHOOSE THE": :
     "CORRECT NUMBER."
2770 PRINT : :"YOU MUST NAME THE BONES"
     : :"CORRECTLY TO CONTINUE.": : :
2780 GOSUB 230
2790 FLAG=2
2800 GOTO 2490
2810 DATA 67,72,79,79,83,69,58,49,32,72
     ,69,65,68,32,50,32,65,82,77,83,32,
     51,32
2820 DATA 84,79,82,83,79,52,32,76,69,71
     ,83,32,53,32,69,78,68,32,32
2830 RESTORE 2810
2840 CALL HCHAR(23,1,32,21)
2850 FOR X=7 TO 17 STEP 2
2860 FOR Y=23 TO 29
2870 READ G
2880 CALL HCHAR(X,Y,G)
2890 NEXT Y
2900 NEXT X
2910 CALL KEY(0,K,S)
2920 IF S<1 THEN 2910
2930 IF K=53 THEN 2990
2940 IF (K>52)+(K<49)=-1 THEN 2910
2950 CALL CLEAR
2960 PRINT "ONE MOMENT PLEASE": : :
2970 CALL COLOR(9,12,1)
2980 ON K-48 GOTO 990,1360,1690,2180
2990 CALL CLEAR
3000 END
```

Computer Techniques for Tutoring the Mentally Handicapped

Huzzah, the revolution has just started! And the fact that you're reading *The Best of 99'er* signifies that you are very much a part of it—a revolution fueled by the availability and affordability of computer power to millions of consumers. As more and more software—computer programs that can meet a large number of everyday needs, as well as solve problems encountered in special areas—is developed, the computer will become as common in our homes as the telephone.

Our task in this generation is to learn to take advantage of this tool in a variety of areas, disciplines and endeavors. In this article, we would like to focus the application of computer technology on what may seem at first to be a most unlikely area—tutoring the developmentally disabled.

Retardation is defined as "below average intellectual functioning that originates during the developmental stages with associated maladaptive behavior." In the search for tools to combat retardation, the microcomputer has shown itself to be extremely valuable by assisting the retarded population to develop skills, abilities, concepts, and even behaviors. Preliminary testing demonstrates that not only can these individuals use a keyboard, but they can learn it very quickly—finding it attractive, novel and magnetic. Options such as the light pen, joystick, and voice synthesizer provide capabilities that can be used to adapt numerous programs for this special population.

Help for the Schools

Of more than eight million handicapped children in the U.S., reportedly only half are receiving appropriate educational services. School districts under ever-tightening budgets struggle to meet the needs of these children. It therefore appears highly probable that using microcomputers to assist in meeting the needs of these children will be both an economic boon to schools, and a valuable enhancement to the learning process of these youngsters.

Despite traditional controversies regarding the learning process, there are some areas of general agreement. These areas have provided us with a basis for software geared to the special learning needs of the retarded—programs utilizing the unique qualities of the computer to further stimulate learning.

Fascination With the Medium

Retarded and non-retarded alike are able to learn more, as well as more easily, from teaching aids that effectively focus their attention on the content. Attention management for the retarded youngster is especially critical. In this regard, the computer, keyboard, and CRT have a fascination that commands attention with an immediacy that is unparalleled. When a youngster is seated before a console, the attraction of the mechanism coupled with the allure of a good interactive program provides an incredible amount of motivation and drive. If you have children who play computer games or other electronic games using a microprocessor, you already know just how difficult it is to distract them and draw their attention to something else—like homework, eating, or cleaning their room.

Nothing Succeeds Like Success

As human beings, we tend to strive toward success or try to avoid failure. In the search for success, the "locus of control" is usually internal. This is to say that in the process of maturing, a person begins to realize a power or ability to control events, and begins to set goals. We begin to become efficient in attaining goals. Actually attaining them brings a sense of success which is its own reward and prompts one to continue to strive for success.

Avoiding failure, on the other hand, means maintaining a mere minimum of effort so as not to incur some type of punishment. Consequently, the locus of control is external. For a majority of developmentally disabled, avoiding punishment becomes the usual way of behaving. They are not given to setting goals since they have not come to experience the internal locus of control and the possibility of success.

With the use of computers, a learning environment can be created which can provide a retarded child or adult with the experience of success. As the experience is repeated, the locus of control begins to shift from without to within. This is a natural reward process which has more lasting effects than punishment or negative reinforcement. As the repertoire is gradually expanded, the retarded individual begins to realize a potential: a power for success.

A Multisensory Lens

Another important element in the learning process of the retarded person is the ability to focus in on significant cues. Once again, the hardware's attractiveness (or novelty, if you will) is so engaging and attention-riveting (thereby limiting external or irrelevant stimuli or signals) that the person learns to be attentive to only the important and discriminating cues. Furthermore, the multisensory impact of the computer provides an additional quality which is extremely valuable in the learning process of the retarded person: The more you can use, engage, and impact many sensory modalities—

and do it repeatedly in an interesting manner—the greater the likelihood of retention and learning.

An Example Program

The following is a simple program designed for teaching retarded persons the extremely abstract concepts of number recognition, counting and subtraction. We feel that the program demonstrates the principles stated in this article, as well as the uniqueness of the computer as a tool especially well-suited to meeting the learning needs of the developmentally disabled. We wish to emphasize that the computer does not totally substitute for a teacher. The retarded individuals on whom we tested the program required personal assistance and encouragement at the beginning of the lesson. Reaction to the computer ranged from reluctance to eager enthusiasm. In some cases, we first used another program (a keyboard trainer) to familiarize the student with the key locations on the console. The TI-99/4 keyboard is highly suited for use by those unfamiliar with typewriters. We found it helpful, however, to cover the letter keys with masking tape to reduce distractions. Also, we noted some confusion created by the shift characters above each number—a small problem we hope to overcome by trying a number of key covers. Based on field testing of this program, we are convinced that this approach can be extended to many areas of work with this group, a group whose needs are so unique that conventional methods have been only moderately successful. Using this technology, we have a potential for far greater success and the possibility of doing things that were unthought-of for this segment of the population.

The Program

The program opens with several options which must be selected. The instructor is informed that a performance rating of the student's progress is available by pressing the AID key. This rating gives the number of trials, correct answers, and percent correct. If you wish to reset the options later, simply press the BACK command and re-enter. The AID and BACK commands can be entered during the main lesson, thus giving the instructor flexibility in choosing the set of options most appropriate to the student's level of ability. The program also has a speech selection option that permits its use without the Speech Synthesizer and *Speech Editor* Command Cartridge. [The extensive use of graphics in this program precludes the use of the speech editor resident in the *Extended BASIC* Command Cartridge with its fewer available character sets.—Ed.] Although the actual lesson is designed for non-readers, the initial option selection must be performed by an instructor or someone who can read. These options can be selected in any combination from the following list:

Select:

1 = Random presentation
2 = Serial presentation
3 = End lesson
—Display the number above the gulls (Y/N)
—Pronounce each number as it is printed (Y/N)
—Computer says press_(number) after a row of gulls is put on the screen (Y/N)

Select format for placing of gulls on the ocean:

1 = Horizontal Row
2 = Diagonal Pattern
3 = Random row placement

After the options are selected, the screen clears and a seascape is painted on the screen. A picture of a deep blue ocean and a steamship liner on the horizon moving toward a tropical island focuses the student's attention immediately. The gulls appear on the water from left to right, and a shark's fin begins to circle the gulls while waiting for the student to press the key representing the number of gulls. If the response is correct, a musical fanfare is played, followed by the computer saying "Right (number)," and the ship moves one column to the left, emitting smoke puffs from the stacks (the number of puffs equal to the number of gulls). However, if the student's response is incorrect, the computer says, "Uh oh," and the shark stops circling the gulls, emerges from the water and devours the last gull (with sound effects)! Then the computer asks the student, "What number is left?" and waits for the student to press the key representing the number of remaining gulls. If incorrect again, the computer says, "That is incorrect," gives a short laugh, and then engulfs the next gull! This can continue until no gulls remain; the program then recycles and another trial begins. On a correct response the computer says "Right (number)" and the ship is advanced one column to the left with the appropriate number of smoke puffs. Each correct response advances the ship toward the island until it is "docked" and the computers says, "You win." It then recycles the program, placing the ship back at the right side of the screen, and continues the lesson.

We recommend that students start with the Serial option rather that the Random. This starts with the number 1 and adds a number on each correct trial, but will not add a number on an error. In this way, a student cannot be challenged by the larger numbers until he has displayed mastery of the smaller ones. In general, we also recommend the strategy of starting a student with all prompt options operating, then removing them as the student demonstrates competence.

EXPLANATION OF THE PROGRAM
Computer Techniques for Tutoring the Mentally Handicapped

Line Nos.	
160-280	Sets all variables to zero.
290-820	Instructor selects program options.
830-1310	Defines characters and color codes.
1320-1450	Constructs seascape, boat, and island.
1460-1550	Calculates the appropriate number of gulls to place on screen.
1560-1590	Clears screen from row 10 to 24.
1600-1820	Places gulls in the water.
1830-1890	Controls movement of shark fin from left to right.
1900-1960	Evaluates key response while shark circles gulls.
1970-2120	Musical fanfare on correct response.
2130-2220	Controls movement of shark fin from right to left background.
2130-2290	Evaluates key response.
2300-2530	Controls animation of shark eating gulls.
2540-2620	Evaluates key response and clears screen to right of last gull after shark "eats" it.
2630-2660	Controls loop to eat next gull.
2670-2740	Verbal response to correct key press; increments number of trials and right responses.
2750-2810	Moves ship, controls puffs of smoke and sound effects from ship stacks.
2920-2950	Prompts to press a number if a letter was pressed.
2960-3060	Routine when boat reaches island.
3070-3100	Calculates performance scores.
3110-3170	Prints option to end and branches appropriately.

```
100 REM  ***********************
110 REM  *  COUNTING  LESSON  *
120 REM  ***********************
130 REM
140 REM
150 REM
160 CALL CLEAR
170 CALL SCREEN(8)
180 REM  SET VARIABLES TO ZERO
190 FOR LOOP=1 TO 10
200 P(LOOP)=0
210 NEXT LOOP
220 T=0
230 DN=0
240 PN=0
250 L=0
260 N=0
270 CT=0
280 NM=0
290 REM  MENUS
300 PRINT "  INSTRUCTOR PLEASE NOTE"
310 PRINT "**********************
*":::
320 PRINT "DURING THE LESSON":"YOU CAN
  PRESS <AID> TO GET":"A READOUT OF
  PERCENT OF":"CORRECT RESPONSES"
330 PRINT :"IF YOU WISH TO RESET PROGR
AM":"OPTIONS PRESS THE <BACK> KEY"
:::
340 PRINT :"DO YOU WANT TO USE SPEECH"
:"OPTION? (Y/N) (SPEECH EDITOR":"A
ND SYNTHESIZER REQUIRED)"
350 CALL KEY(0,K,S)
360 IF S=0 THEN 350
370 IF K=89 THEN 430
380 IF K<>78 THEN 350
390 SP=1
400 PN=1
410 CT=1
420 GOTO 440
430 SP=0
440 PRINT ::::::"  SELECT FROM THE FOLL
OWING":"**********************
**":::
450 PRINT TAB(8);"1-RANDOM GROUPS":TAB
(8);"2-SEQUENTIAL":TAB(10);"PRESEN
TATION"
460 PRINT TAB(8);"3-END LESSON"
470 CALL KEY(0,K,S)
480 IF S=0 THEN 470
490 IF K=50 THEN 530
500 IF K=51 THEN 3070
510 IF K<>49 THEN 470
520 L=1
530 PRINT :"DO YOU WANT TO DISPLAY THE
":"NUMBERS ABOVE THE GULLS?":"(Y/N
)":::
540 CALL KEY(0,K,S)
550 IF S=0 THEN 540
560 IF K=89 THEN 590
570 IF K<>78 THEN 540
580 DN=1
590 IF SP=1 THEN 720
600 PRINT "DO YOU WANT TO PRONOUNCE":"
EACH NUMBER AS IT IS PRINT-":"ED?
  (Y/N)":::
610 CALL KEY(0,K,S)
620 IF S=0 THEN 610
630 IF K=89 THEN 660
640 IF K<>78 THEN 610
650 PN=1
660 PRINT "DO YOU WANT THE COMPUTER TO
":"TELL THE STUDENT WHICH KEY":"TO
  PRESS FOR EACH TRIAL?":"(Y/N)"
670 CALL KEY(0,K,S)
680 IF S=0 THEN 670
690 IF K=89 THEN 720
700 IF K<>78 THEN 670
710 CT=1
720 PRINT :::"DO YOU WANT THE GULLS TO
BE":"PRESENTED:":"  1-ALL ON SAME
  LINE":"  2-DIAGONAL":"  3-RANDO
M"
730 CALL KEY(0,K,S)
740 IF S=0 THEN 730
750 IF K=49 THEN 820
760 IF K=50 THEN 800
770 IF K<>51 THEN 730
780 LN=2
790 GOTO 840
800 LN=1
810 GOTO 840
820 LN=0
830 REM  DEFINE CHARACTERS
840 RANDOMIZE
850 CALL CLEAR
860 CALL SCREEN(8)
870 A$="E0F0F8FCFEFFFFFB"
880 B$="7F3F1F0F070FFFFF"
890 C$="000000000000080C0"
900 D$="E0F0F8FCFEFFFFFF"
910 E$=""
920 F$="0103070F1F3F7FFF"
930 G$="0001010101030707F"
940 H$="8080800F0F8FCFEFF"
950 I$="FFF7FFBAFFBFEBFF"
960 J$="3337F7FF5FFAFFFF"
970 K$="000000003F1F0F07"
980 L$="FFAAFFFFFFFFFFFF"
990 M$="0000000FFFFFFFEFC"
1000 N$="00000000C3C3C3C3"
1010 O$="0000000000081422"
1020 P$="000E0F0C86FE7E3C"
1030 Q$=""
1040 R$="418200C300C3C3C3"
1050 S$=""
1060 CALL CHAR(96,A$)
1070 CALL CHAR(97,B$)
1080 CALL CHAR(98,C$)
1090 CALL CHAR(99,D$)
1100 CALL CHAR(104,E$)
1110 CALL CHAR(100,F$)
1120 CALL CHAR(101,G$)
1130 CALL CHAR(112,H$)
1140 CALL CHAR(113,I$)
1150 CALL CHAR(114,J$)
1160 CALL CHAR(120,K$)
1170 CALL CHAR(121,L$)
1180 CALL CHAR(122,M$)
1190 CALL CHAR(128,N$)
1200 CALL CHAR(136,O$)
1210 CALL CHAR(102,P$)
1220 CALL CHAR(140,Q$)
1230 CALL CHAR(144,R$)
1240 CALL CHAR(145,S$)
1250 CALL COLOR(14,8,6)
1260 CALL COLOR(9,16,6)
1270 CALL COLOR(11,13,11)
1280 CALL COLOR(10,11,11)
1290 CALL COLOR(12,7,11)
1300 CALL COLOR(13,2,11)
1310 CALL COLOR(15,16,11)
1320 CALL HCHAR(1,1,104,288)
1330 CALL HCHAR(10,1,140,480)
1340 CALL HCHAR(9,10,112)
1350 CALL HCHAR(9,1,113,9)
1360 CALL HCHAR(8,9,112)
1370 CALL HCHAR(8,1,113,8)
1380 CALL HCHAR(7,8,112)
1390 CALL HCHAR(7,1,113,7)
1400 CALL HCHAR(6,7,112)
1410 CALL HCHAR(6,1,114,6)
1420 CALL HCHAR(9,27-X,120)
1430 CALL HCHAR(9,28-X,121,3)
1440 CALL HCHAR(9,31-X,122)
1450 CALL HCHAR(8,29-X,128)
1460 REM  CALCULATE NUMBER OF GULLS
1470 IF L<>0 THEN 1540
```

```
1480 IF T=1 THEN 1540
1490 N=N+1
1500 IF N>9 THEN 1540
1510 GOTO 1570
1520 N=1
1530 GOTO 1570
1540 N=INT(RND*9)+1
1550 T=1
1560 REM  CLEAR 'WATER' AREA
1570 FOR ERASE=10 TO 24
1580 CALL HCHAR(ERASE,1,140,32)
1590 NEXT ERASE
1600 REM  LOOP PLACES GULLS IN WATER
1610 FOR J=1 TO (2*N) STEP 2
1620 NM=NM+1
1630 REM  CALCULATE POSITION OF GULLS
1640 IF LN=0 THEN 1690
1650 IF LN=1 THEN 1680
1660 P(NM)=(RND*8)+1
1670 GOTO 1690
1680 P(NM)=J/2
1690 CALL HCHAR(12+P(NM),J+9,136)
1700 IF DN=1 THEN 1720
1710 CALL HCHAR(11+P(NM),J+10,ASC(STR$(
     NM)))
1720 CALL HCHAR(12+P(NM),J+10,102)
1730 IF PN=1 THEN 1750
1740 CALL SAY(STR$(NM))
1750 NEXT J
1760 C=1
1770 CALL HCHAR(10,1,140,32)
1780 FS="80C0E0F0F8FCFEFF"
1790 CALL CHAR(100,FS)
1800 IF CT=1 THEN 1840
1810 CALL SAY("PRESS")
1820 CALL SAY(STR$(NM))
1830 REM  START SHARK FIN ACROSS SCREEN
     IN FOREGROUND
1840 CALL HCHAR(22,C,100)
1850 CALL SOUND(25,110,0)
1860 CALL HCHAR(22,C,136)
1870 C=C+1
1880 FB=0
1890 IF C=32 THEN 2140
1900 CALL KEY(0,K,STATUS)
1910 IF STATUS=0 THEN 1840
1920 IF K=1 THEN 3080
1930 IF K=15 THEN 160
1940 IF K<48 THEN 2920
1950 IF K>57 THEN 2920
1960 NN=K-48
1970 IF NN<>NM THEN 2330
1980 NM=0
1990 REM  PLAY MUSICAL FLOURISH ON CORR
     ECT RESPONSE
2000 CALL SOUND(100,349,0)
2010 CALL SOUND(100,440,0)
2020 CALL SOUND(100,523,0)
2030 CALL SOUND(100,523,0)
2040 CALL SOUND(100,523,0)
2050 CALL SOUND(100,440,0)
2060 CALL SOUND(100,440,0)
2070 CALL SOUND(100,440,0)
2080 CALL SOUND(100,349,0)
2090 CALL SOUND(100,440,0)
2100 CALL SOUND(100,349,0)
2110 CALL SOUND(100,262,0)
2120 GOTO 2690
2130 REM  START SHARK FIN ACROSS SCREEN
     IN BACKGROUND
2140 FS="0000001030707F1F3F"
2150 CALL HCHAR(22,1,140,32)
2160 CALL CHAR(100,FS)
2170 CALL HCHAR(10,C,100)
2180 CALL SOUND(25,110,5)
2190 CALL HCHAR(10,C,136)
2200 IF C=1 THEN 1760
2210 C=C-1
2220 FB=1
2230 CALL KEY(0,K,STATUS)
2240 IF STATUS=0 THEN 2170
2250 IF K=1 THEN 3080
2260 IF K=15 THEN 160
2270 IF K<48 THEN 2920
2280 IF K>57 THEN 2920
2290 NN=K-48
2300 IF NN<>N THEN 2330
2310 GOTO 1980
2320 REM  SHARK EATS GULL ON WRONG ANS
     WER
2330 R=10
2340 IF SP=1 THEN 2360
2350 CALL SAY("UHOH")
2360 TRIAL=TRIAL+1
2370 FS="FF7E3C1800000000"
2380 CALL CHAR(100,FS)
2390 FOR I=9+(2*N) TO 10 STEP -2
2400 FOR J=1 TO 2
2410 BS="7F3F3F3F1F0F0703"
2420 CALL CHAR(97,BS)
2430 CALL HCHAR(11+P(NM),I,96)
2440 CALL HCHAR(11+P(NM),I+1,98)
2450 CALL HCHAR(12+P(NM),I,97)
2460 CALL HCHAR(12+P(NM),I+1,99)
2470 CALL HCHAR(13+P(NM),I+3,100)
2480 CALL SOUND(50,-5,0)
2490 BS="7F3F1F0F070FFFFF"
2500 CALL CHAR(97,BS)
2510 NEXT J
2520 NM=NM-1
2530 N=N-1
2540 REM  AFTER EATING GULL/WAITS FOR C
     ORRECT ANSWER FOR NUMBER LEFT
2550 IF SP=1 THEN 2570
2560 CALL SAY("WHAT+NUMBER+IS+LEFT")
2570 CALL KEY(0,K,S)
2580 IF S=0 THEN 2570
2590 FOR LOOP=10 TO 24
2600 CALL HCHAR(LOOP,I,140,32-I)
2610 NEXT LOOP
2620 IF K=ASC(STR$(N))THEN 2690
2630 TRIAL=TRIAL+1
2640 IF SP=1 THEN 2660
2650 CALL SAY("#THAT IS INCORRECT# A1+A
     1+A1")
2660 NEXT I
2670 NM=0
2680 IF N<1 THEN 1470
2690 IF SP=1 THEN 2720
2700 CALL SAY("RIGHT")
2710 CALL SAY(CHR$(K))
2720 RIGHT=RIGHT+1
2730 TRIAL=TRIAL+1
2740 NM=0
2750 REM  MOVES SHIP ACROSS SCREEN ONE
     COLUMN
2760 X=X+1
2770 IF X=17 THEN 2970
2780 CALL HCHAR(9,11,145,21)
2790 CALL HCHAR(8,11,145,21)
2800 CALL HCHAR(9,27-X,120)
2810 CALL HCHAR(9,28-X,121,3)
2820 CALL HCHAR(9,31-X,122)
2830 CALL HCHAR(8,29-X,128)
2840 FOR I=1 TO N
2850 CALL HCHAR(7,29-X,144)
2860 CALL SOUND(200,-5,0)
2870 FOR D=1 TO 50
2880 NEXT D
2890 CALL HCHAR(7,29-X,145)
2900 NEXT I
2910 GOTO 1470
2920 IF SP=1 THEN 2940
2930 CALL SAY("PRESS+A+NUMBER+PLEASE")
2940 TRIAL=TRIAL+1
2950 IF FB=1 THEN 2170 ELSE 1840
2960 REM  ROUTINE WHEN BOAT REACHES ISL
     AND
```

```
2970 X=0
2980 IF SP=1 THEN 3000
2990 CALL SAY("*YOU WIN*")
3000 CALL SOUND(2000,220,0)
3010 CALL SOUND(50,220,30)
3020 CALL SOUND(1000,220,0)
3030 CALL SOUND(50,220,30)
3040 CALL SOUND(500,220,0)
3050 CALL CLEAR
3060 GOTO 1320
3070 REM  CALCULATE SCORES
3080 CALL CLEAR
```

```
3090 NM=0
3100 PRINT "TRIALS=";TRIAL::"RIGHT=";RI
     GHT::"PERCENT CORRECT=";INT(100*(R
     IGHT/TRIAL))
3110 PRINT ::"TO RETURN TO LESSON PRESS
     1":"TO END LESSON PRESS 2"::
3120 CALL KEY(0,K,S)
3130 IF S=0 THEN 3120
3140 IF K=49 THEN 850
3150 IF K<>50 THEN 3120
3160 PRINT "SO LONG FOR NOW!"
3170 END
```

Typing for Accuracy

TI BASIC

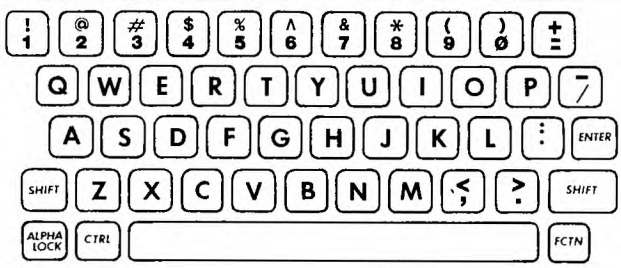
Typing for Accuracy provides practice for students who are somewhat familiar with the keyboard.

Seven finger-placement categories using different typewriter keys are offered: home keys; home row; top row, middle finger; top row, pointer finger; ring finger; little finger; and bottom row. A typist may choose one of the categories for each drill.

The program uses graphics and sound effects to liven up the drill: A rocket appears on the screen, and a word is printed on the rocket while a 1.5-second tone sounds. A student then types and ENTERs the word. If it has been typed incorrectly, the rocket blasts; if it has been typed correctly, a second tone sounds and the score goes up. The rocket then takes off (with gases trailing behind), and a different word appears.

At the end of ten words the student's score is tabulated and displayed as a percent accuracy rating. The student may then choose from the seven drills or may exit the program.

This drill is not meant to be a speed drill because beginning typing students must gain accuracy and familiarity with the keyboard before working on speed. However, if the student wants a timed test, an approximate words-per-minute rate can be estimated using the tones—i.e., if the student presses ENTER as the tone ends, the rate is 40 wpm.



EXPLANATION OF THE PROGRAM
Typing for Accuracy

Line Nos.	
120	Dimensions the array A$ to allow for twenty words.
130	Sets the y-coordinate for drawing the rocket.
140-200	Words used in the drills.
210	Prints the title screen.
220	Prints instructions.
230	Prints the menu screen of the seven categories.
240-400	Awaits the student's choice. Depending on the category chosen, a certain DATA statement is RESTOREd which contains the words for that particular category.
410-440	Draws the rocket.
450-480	Reads the number of words in the category and stores the words in the A$ array.
490	Initializes the score.
500-530	Randomly chooses a word. Once a word is chosen it is not used again in the drill.
540	Calculates the coordinate for printing the word.
550	Sounds the tone for 1.5 seconds.
560-580	Prints the word to be typed on the rocket.
590	Awaits the student's typed word.
600-640	Compares the student's word with the given word. If it is incorrect, a white noise is sounded; if it is correct, a tone sounds and the score is incremented.
650-700	If it is the first word, draws the bottom of the rocket.
710-730	If it is the second word, completes the fins of the rocket.
740-780	The rocket moves up and has a trail under it. The words are cleared.
790	A$ set to zero so the word cannot be used again.
800	Returns for the next word.
810-890	Prints score and waits until student is ready to continue.
910	END.
Subroutines:	
920-1530	Prints title screen with music.
1540-1680	Prints instructions.
1690-1780	Prints menu screen of seven categories.
1700-2110	Draws the rocket.

```
100 REM  * TYPING FOR ACCURACY *
110 REM
120 DIM A$(20)
130 Y=20
140 DATA 15,FAD,A,AS,DAD,AD,SAD,LAD,FA
    LL,ALFALFA,SASS,LASS,DADS,LADS,FAL
    LS,FADS
150 DATA 16,HAD,HAS,GAS,SAG,HALL,HALLS
    ,LADS,SAGS,HAG,LAG,LAGS,SLAG,SHALL
    ,SASH,DASH,FLASH
160 DATA 17,DEAF,DEED,SEED,FEED,HEED,L
    IKE,KILL,FILL,FEEL,FEES,LIED,DIAL,
    SLIDE,FLIES,LIFE,SLID,GLIDE
170 DATA 17,TAG,HAT,TALL,THY,DAY,HAY,J
    AY,GAY,LAY,TAR,RAT,STAR,STAFF,FAST
    ,TRY,SAY,YARD
180 DATA 20,WISH,EXAM,EXACT,TEXT,TWO,W
    ON,SOW,WASH,WORLD,OWE,WORD,LOOK,LO
    SE,SOD,WOW,TOW,TEXAS,OXEN,MIX,WORS
    E
190 DATA 16,QUAKE,QUIZ,QUIP,ZAP,QUIT,P
    IQUE,PLAQUE,PUZZLE,PLAZA,SAP,ZIPPE
    R,PRIZE,QUICK,SQUEEZE,ZEAL,ZIP
200 DATA 18,CALM,CAN,MEN,NIMBLE,EXACT,
    EXAM,MIX,NIX,BUZZ,ZOOM,NAVY,CAB,BA
    CK,BOMB,ZOMBIE,CAVE,VACATE,VARMINT
210 GOSUB 920
220 GOSUB 1540
230 GOSUB 1690
240 CALL KEY(0,KEY,STS)
250 IF KEY<49 THEN 240
260 IF KEY>56 THEN 240
270 ON (KEY-48)GOTO 280,300,320,340,36
    0,380,400,910
280 RESTORE 140
290 GOTO 410
300 RESTORE 150
310 GOTO 410
320 RESTORE 160
330 GOTO 410
340 RESTORE 170
350 GOTO 410
360 RESTORE 180
370 GOTO 410
380 RESTORE 190
390 GOTO 410
400 RESTORE 200
410 CALL CLEAR
420 CALL COLOR(1,2,1)
430 CALL SCREEN(8)
440 GOSUB 1890
450 READ N
460 FOR I=1 TO N
470 READ A$(I)
480 NEXT I
490 SCORE=0
500 RANDOMIZE
510 FOR K=1 TO 10
520 W=INT(N*RND)+1
530 IF A$(W)="0" THEN 520
540 XC=24-K
550 CALL SOUND(15000,-1,2)
560 FOR J=1 TO LEN(A$(W))
570 CALL HCHAR(XC,J+17,ASC(SEG$(A$(W),
    J,1)))
580 NEXT J
590 INPUT B$
600 IF B$=A$(W)THEN 630
610 CALL SOUND(1000,-7,1)
620 GOTO 650
630 CALL SOUND(1000,-2,1)
640 SCORE=SCORE+1
650 IF K<>1 THEN 710
660 CALL HCHAR(23,Y-4,98)
670 CALL HCHAR(23,Y+4,98)
680 CALL HCHAR(23,Y-3,99)
690 CALL HCHAR(23,Y+3,100)
700 GOTO 740
710 IF K<>2 THEN 740
720 CALL HCHAR(23,Y-4,99)
730 CALL HCHAR(23,Y+4,100)
740 CALL HCHAR(23,Y-1,105,3)
750 CALL SOUND(1,44000,30)
760 CALL HCHAR(XC-1,Y-2,98,7)
770 CALL HCHAR(XC-1,Y+5,32,3)
780 CALL HCHAR(23,1,32,15)
790 A$(W)="0"
800 NEXT K
810 CALL CLEAR
820 FOR I=2 TO 8
830 CALL COLOR(I,2,1)
840 NEXT I
850 SC=10*SCORE
860 PRINT :::::"YOUR SCORE IS";SC,"PER
    CENT ACCURACY."
870 PRINT ::::"PRESS ENTER TO CONTINUE.
    "
880 CALL KEY(0,KEY,ST)
890 IF KEY<>13 THEN 880
900 GOTO 230
910 END
920 CALL CLEAR
930 CALL SCREEN(5)
940 T=500
950 CALL SOUND(T,880,3,698,8,294,10)
960 PRINT :::TAB(11);"T Y P I N G"
970 CALL CHAR(104,"FFFF00FFF00FFFF")
980 CALL SOUND(T,932,3,784,8,196,11)
990 PRINT ::TAB(15);"FOR"
1000 CALL COLOR(10,16,6)
1010 CALL COLOR(11,13,1)
1020 CALL SOUND(T,784,3,659,8,262,10)
1030 PRINT ::TAB(12);"ACCURACY"
1040 CALL CHAR(112,"00E0F8FEFFFFFFFF")
1050 CALL SOUND(T,880,3,698,8,175,12)
1060 CALL CHAR(113,"0000000000080E0F8FE")
1070 CALL CHAR(114,"80E0FCFFFFFCE08")
1080 CALL CHAR(115,"FEF8E08")
1090 CALL SOUND(T,698,3,587,8,233,10)
1100 PRINT :::::::::
1110 PRINT :::
1120 CALL CHAR(116,"FFFFFFFFFFEF8E")
1130 CALL SOUND(T/2,784,3,165,10)
1140 CALL CHAR(96,"80C0E0F8F8E0C08")
1150 CALL SOUND(T/2,698,3,165,10)
1160 CALL CHAR(97,"80C0E0F0F8FCFEFF")
1170 CALL SOUND(T/2,659,3,277,10)
1180 CALL CHAR(98,"FFFFFFFFFFFFFFFF")
1190 CALL CHAR(117,"FFFFFFFFFFFFFFFF")
1200 CALL SOUND(T/2,784,3,277,10)
1210 CALL CHAR(99,"FFFEFCF8F0E0C08")
1220 CALL SOUND(T*2,698,2,587,8,147,12)
1230 CALL HCHAR(15,7,117,7)
1240 CALL HCHAR(16,7,117,9)
1250 CALL HCHAR(17,7,117,7)
1260 CALL HCHAR(16,16,114)
1270 CALL HCHAR(15,15,113)
1280 CALL HCHAR(17,15,115)
1290 CALL HCHAR(15,14,112)
1300 CALL HCHAR(17,14,116)
1310 CALL SOUND(T,466,4,165,10)
1320 FOR XX=15 TO 17
1330 CALL HCHAR(XX,1,104,6)
1340 NEXT XX
1350 CALL SOUND(T,440,3,175,10)
1360 CALL COLOR(2,16,1)
1370 FOR YY=18 TO 26 STEP 2
1380 CALL HCHAR(16,YY,42)
1390 NEXT YY
1400 CALL SOUND(T,698,3,440,8,294,10)
1410 CALL HCHAR(16,28,49)
1420 CALL HCHAR(16,29,48,2)
1430 CALL HCHAR(16,31,37)
1440 CALL SOUND(T,784,3,587,8,233,10)
1450 CALL CHAR(100,"7F3F1F0F07030301")
1460 CALL CHAR(101,"000103070F1F3F7F")
1470 CALL SOUND(T/2,698,3,392,8,262,10)
1480 CALL CHAR(102,"0000181818C3C7EFF")
```

```
1490 CALL SOUND(T/2,659,2,262,10)
1500 CALL CHAR(105,"DBDBDBDBDBDBDBDB")
1510 CALL SOUND(4*T,698,2,440,8,175,10)
1520 CALL SOUND(1,44000,30)
1530 RETURN
1540 CALL CLEAR
1550 CALL SCREEN(7)
1560 CALL COLOR(2,2,1)
1570 PRINT "PICK A TYPING CATEGORY."
1580 PRINT :::"YOU WILL SEE A WORD"
1590 PRINT "IN THE ROCKET."
1600 PRINT :"TYPE AND ENTER IT BEFORE T
HE TONE ENDS."
1610 PRINT :"IF YOU ARE CORRECT,"
1620 PRINT "ANOTHER TONE SOUNDS,"
1630 PRINT :"IF YOU ARE INCORRECT."
1640 PRINT "YOU WILL BE BLASTED.":::::
1650 PRINT "PRESS ENTER TO CONTINUE."
1660 CALL KEY(0,KEY,ST)
1670 IF KEY<>13 THEN 1660
1680 RETURN
1690 CALL CLEAR
1700 CALL SCREEN(12)
1710 CALL COLOR(1,2,12)
1720 FOR I=2 TO 8
1730 CALL COLOR(I,1,12)
1740 NEXT I
1750 PRINT :::::" CHOOSE ONE":::
1760 PRINT :" 1 HOME KEYS"
1770 PRINT :" 2 HOME ROW"
1780 PRINT :" 3 TOP ROW, MIDDLE FINGER
"
1790 PRINT :" 4 TOP ROW, POINTER FINGE
R"
1800 PRINT :" 5 RING FINGER"
1810 PRINT :" 6 LITTLE FINGER"
1820 PRINT :" 7 BOTTOM ROW"
1830 PRINT :" 8 END PROGRAM":::
1840 CALL SCREEN(5)
1850 FOR I=2 TO 8
1860 CALL COLOR(I,2,12)
1870 NEXT I
1880 RETURN
1890 CALL COLOR(9,1,1)
1900 CALL COLOR(10,1,1)
1910 CALL VCHAR(12,Y,98,13)
1920 CALL VCHAR(13,Y-1,98,12)
1930 CALL VCHAR(13,Y+1,98,12)
1940 CALL VCHAR(14,Y-2,98,11)
1950 CALL VCHAR(14,Y+2,98,11)
1960 CALL VCHAR(13,Y-2,101)
1970 CALL VCHAR(13,Y+2,97)
1980 CALL VCHAR(12,Y-1,101)
1990 CALL VCHAR(12,Y+1,97)
2000 CALL VCHAR(11,Y,102)
2010 CALL VCHAR(22,Y-3,98,3)
2020 CALL VCHAR(22,Y+3,98,3)
2030 CALL VCHAR(23,Y-4,98,2)
2040 CALL VCHAR(23,Y+4,98,2)
2050 CALL VCHAR(22,Y-4,101)
2060 CALL VCHAR(22,Y+4,97)
2070 CALL VCHAR(21,Y-3,101)
2080 CALL VCHAR(21,Y+3,97)
2090 CALL COLOR(9,7,1)
2100 CALL COLOR(10,16,1)
2110 RETURN
2120 END
```

CIVIL ENGINEERING

SIMPLE BEAMS

TI BASIC

The purpose of this program is to tutor civil engineering students who are studying statics or structures. It is limited to a simple determinate beam supported at the ends and loaded with a concentrated load, a uniform load, or a combination of a concentrated load and a uniform load. Basic knowledge of elementary statics is a prerequisite.

1. Concentrated load at the center

Newton's laws of force and moments are reviewed. The general solution of a load P applied at the center of a beam of length L is developed for the reaction forces A and B at each end of the beam. The student then does two problems. The load P and the length L are chosen randomly for the problems. If the student enters an incorrect solution, the correct one is given, and he is given another problem,

2. Concentrated load anywhere

The general solution of a load P applied a distance D from end A on a beam of length L is derived for the reaction forces A and B at each end. An example problem is given and solved. Then a problem is given for which the student enters his answers. The program prints the method of solution. For the next problem the student enters his solution. If he is incorrect, the program shows him how to solve the problem, and he is given another problem to solve.

3. Uniform load

The uniform load is considered as an equivalent concentrated load acting at the centroid of the loading pattern. The first example is a uniform load for the length of the beam and is solved in general terms. The student is then given a problem. If he enters an incorrect answer, he is shown the correct solution and given another problem.

4. Combination loads

Instructions are provided for placing a beam with one concentrated load and one uniform load. The student is then given a problem with combination loads chosen randomly. The program draws and labels the beam for each problem. If the student enters an incorrect solution, the correct solution is printed and he is given another problem.

5. Problems

No instruction is given. The program randomly chooses a beam length and loading pattern, and prints the problem. It then draws and labels the beam. The student enters his answers; if he is incorrect, the correct answers are given and another problem is printed.

6. Your own problem

The student enters the beam length and loading specifications. The program computes the reaction forces A and B at the ends.

After each section has been completed with correct solutions, the student is given the choice of having more of the same kind of problems, entering his own problems, or returning to the menu screen.

Programming Techniques

This program is a teaching aid or tutor, so it incorporates pauses, allowing the student to work on the problem before continuing. The student must enter a correct solution to the problem before he or she can go on to a different kind of problem. If the student enters an incorrect solution, the correct answers are printed and another problem of the same type is presented.

The numbers for each problem are chosen randomly (yet appropriately) for each beam. The length of the beam is between 10 and 20 feet. The concentrated load is 100 times a random number from one to twenty (i.e., 100 to 2000 pounds), and is placed at a distance D from end A (randomly chosen within the bounds of the length of the beam).

The uniform load is 10 times a random number from one to ten (i.e.,to 100 pounds per foot). For some of the problems, the uniform load is acting over the length of the beam.

For more advanced problems, it acts between two endpoints, L1 and L2, given as distances from end A. L1 must be equal to or greater than zero and less than the total length of the beam. L2 must be greater than L1 and less than or equal to the total length of the beam.

The problems are written in "story problem" form by using PRINT statements in subroutines, with the program using only the statements that are necessary for each loading condition.

After the student has had time to draw and label the problem on his own paper, the computer will ask him to "PRESS ENTER TO CONTINUE", and the beam will be drawn on the screen with approximate proportions.

The general beam is drawn with a pin at end A and a roller at end B.

The distances are approximated by using a variable y-coordinate—an integer value of the fraction of the distance (D or L1) divided by the total beam length multiplied by the number of characters printed in the general beam. For example, Statement 6750 is:

$$Y = INT(L1/LL*22) + 6$$

Y is the y-coordinate used in CALL HCHAR or CALL VCHAR statements. And 6 is the displacement of the end of the beam from the left side of the screen.

In statement 6760:

$$Z = INT(D2/LL*22) - 1$$

Z is the number of characters to be printed horizontally for the uniform load. D2 is the distance L2 – L1.

Figure 1.

LB$ = STR$(PP)	Converts PP to a string variable.
For II = 1 TO LEN(LB$)	LEN finds the length of LB$.
JJ = II + J – 4	Calculation for y-coordinate.
CALL HCHAR(I – 5,JJ,ASC (SEG$(LB$,II,1)))	Prints each digit in order.
NEXT II	
CALL HCHAR(I – 5,JJ + 1,32)	Prints a space after last digit.
CALL HCHAR(I – 5,JJ + 2,76)	Prints L.
CALL HCHAR(I – 5,JJ + 3,66)	Prints B.
CALL HCHAR(I – 5,JJ + 4,83)	Prints S.

The labels for the values on the beam are variable and are printed using string variables. For example, the concentrated load P may be three or four digits long (100 pounds to 2000 pounds) in the written problems, but the student may input an even longer number. This label printed by using Statements 5850-5930 (see Figure 1).

```
100 REM **  CIVIL  ENGINEERING  **
110 REM
120 REM
130 REM
140 REM
150 REM
160 REM
170 CALL  CLEAR
180 PRINT  TAB(7);"CIVIL  ENGINEERING"
190 PRINT  :  :TAB(9);"FUNDAMENTALS"
200 CALL  COLOR(2,1,1)
210 PRINT  :  :  :  :  :  :TAB(7);"**********
    **********"
220 PRINT  TAB(7);"*                    *"
230 PRINT  TAB(7);"*  SIMPLE  BEAMS     *"
240 PRINT  TAB(7);"*                    *"
250 PRINT  TAB(7);"**********************":  :
    :  :  :
260 FOR  E=1  TO  10
270 CALL  COLOR(2,6,5)
```

```
280 FOR  DELAY=1  TO  75
290 NEXT  DELAY
300 CALL  COLOR(2,5,6)
310 FOR  DELAY=1  TO  75
320 NEXT  DELAY
330 NEXT  E
340 CALL  CLEAR
350 CALL  COLOR(2,2,1)
360 CALL  COLOR(9,2,1)
370 CALL  COLOR(12,11,1)
380 CALL  CHAR(120,"0F0F0F0F0F0F0F0F0F0F")
390 CALL  CHAR(121,"FFFFFFFFFFFFFFFFFFFF")
400 CALL  CHAR(122,"F0F0F0F0F0F0F0F0F0F0")
410 CALL  CHAR(99,"18242424244242818181")
420 CALL  CHAR(100,"1824428181422418")
430 CALL  CHAR(101,"0F091224499")
440 CALL  CHAR(102,"FF2449924")
450 CALL  CHAR(103,"F89020408")
460 CALL  CHAR(104,"10103838545492922")
470 CALL  CHAR(105,"10101010101010101010")
```

EXPLANATION OF THE PROGRAM
Civil Engineering Fundamentals

Line Nos.	
100-250	Prints the title screen.
250-330	Blinks a blue border.
340	Clears the screen.
350-540	Defines special graphics characters for drawing the beam and loading, and sets the color for them.
550-680	Prints second screen, diagram of simple beam.
690	Goes to menu screen for choice of problems.
700	Choices 1 and 2, concentrated loads, branch to here.
710-810	Prints instruction screen.
820-900	Prints second instruction screen.
910	For choice 2, branches to 1720.
920-1000	Prints problem.
1010-1070	Draws and labels general beam.
1080-1180	Shows solution of reaction forces in general terms.
1190-1270	Draws and labels beam with centrally-applied load.
1280-1330	General statement for central load.
1340-1370	Chooses random numbers for problem.
1380-1400	Writes the problem.
1410-1440	Draws and labels the beam.
1450	Asks for A and B from student.
1460-1540	Compares student's answers with calculated solution and prints appropriate remark.
1550-1580	Has another problem.
1590-1610	Asks if student wants more problems and branches accordingly.
1620-1700	Draws and labels a beam for student's problem.
1720-1790	Prints instructions for second type of beam, concentrated load anywhere.
1800-1870	Draws and labels beam.
1880-1970	Solves the problem.
1980-2050	Chooses a problem and prints it.
2060-2160	Draws and labels the beam.
2170-2190	Solves the problem.
2210-2240	Compares input answers with calculated solution.
2250-2270	If student is incorrect, solves the problem in detail.
2280-2290	Returns for another problem.
2300-2330	Solution was correct. If it is the second problem, another problem is chosen.
2340-2360	Offers the student the choice for more problems.
2370-2530	Solves a problem the student enters.
2540-2600	Prints the general problem for a uniform load.

```
480 CALL CHAR(106,"92925454383810 10")
490 CALL CHAR(112,"FFFFFFFFFFFFFFFF")
500 CALL CHAR(113,"F0F0F0F0F0F0F0F0")
510 CALL CHAR(114,"0F0F0F0F0F0F0F0F")
520 CALL CHAR(115,"FF")
530 CALL CHAR(98,"FF60180608304 0FF")
540 CALL COLOR(11,6,1)
550 PRINT TAB(8);"SIMPLE BEAM":::
560 PRINT TAB(5);"SUPPORTED AT ENDS":
    :::::::::::
570 I=17
580 GOSUB 5380
590 J=12
600 GOSUB 5560
610 CALL HCHAR(I-5,12,80)
620 FOR L=I-3 TO I-1
630 CALL HCHAR(L,17,112,10)
640 NEXT L
650 CALL VCHAR(I-3,27,113,3)
660 CALL HCHAR(I-2,19,87)
670 PRINT "  FIND THE REACTION FORCES"
680 GOSUB 5530
690 GOTO 5000
700 CALL CLEAR
710 PRINT "NEWTON'S LAWS"
720 PRINT "ARE NECESSARY TO"
730 PRINT "SOLVE REACTION PROBLEMS.":
    :
740 PRINT "1. EQUILIBRIUM OF FORCES"
750 PRINT "          SUM OF FORCES = 0"
760 PRINT "          SUM OF MOMENTS = 0"
770 PRINT :"2. FORCES ALWAYS OCCUR IN"
780 PRINT "   PAIRS OF EQUAL AND"
790 PRINT "   OPPOSITE FORCES;"
800 PRINT "   ACTION = REACTION":::
810 GOSUB 5590
820 CALL CLEAR
830 PRINT "TO SOLVE A PROBLEM:"::
840 PRINT "DRAW AND LABEL THE PROBLEM.
    "
850 PRINT :"WITH 2 UNKNOWN REACTIONS,"
860 PRINT "SOLVE 2 EQUATIONS:":::
870 PRINT "   SUM OF MOMENTS = 0"
880 PRINT "   SUM OF FORCES = 0"
890 PRINT :"USE CORRECT UNITS.":::
900 GOSUB 5590
910 IF CHOICE=2 THEN 1720
920 CALL CLEAR
930 PRINT "PROBLEM:"
940 PRINT ::"GIVEN A SIMPLE BEAM"
950 PRINT "SUPPORTED AT THE ENDS."
960 PRINT :"IT IS LENGTH L."
970 PRINT :"A CONCENTRATED LOAD P"
980 PRINT "IS AT THE CENTER."
990 PRINT :"IGNORE WEIGHT OF THE BEAM.
    "
1000 GOSUB 5810
1010 I=16
1020 GOSUB 5370
1030 CALL HCHAR(I+1,12,76)
1040 J=16
1050 GOSUB 5560
1060 CALL HCHAR(I-5,16,80)
1070 GOSUB 6040
1080 PRINT "TAKING MOMENTS AT A,":
1090 PRINT "P*L/2 - B*L = 0"
1100 PRINT "          B*L = P*L/2"
1110 PRINT TAB(11);"B = P/2":
1120 PRINT :"NOW TAKE SUM OF FORCES=0"
1130 GOSUB 5590
1140 PRINT "A + B - P = 0"
1150 PRINT "     A + B = P"
1160 PRINT "         A = P - B = P - P/2
     "
1170 PRINT TAB(9);"A = P/2"
1180 GOSUB 5590
1190 GOSUB 5370
1200 CALL HCHAR(I+1,12,76)
1210 GOSUB 5560
1220 CALL HCHAR(11,16,80)
1230 FOR Y=4 TO 26 STEP 22
1240 CALL HCHAR(22,Y,80)
1250 CALL HCHAR(22,Y+1,47)
1260 CALL HCHAR(22,Y+2,50)
1270 NEXT Y
1280 PRINT "IF THE CONCENTRATED LOAD"
1290 PRINT "IS IN THE CENTER,"
1300 PRINT "A = B = P/2":::
1310 PRINT "FOR EXAMPLE, IF P=1000 LBS.
     "
1320 PRINT "     A=500 LBS. AND B=500 LBS
     "
1330 GOSUB 5590
1340 RANDOMIZE
1350 EX=2
1360 PP=100*(INT(20*RND)+1)
1370 LL=INT(6*RND)+10
1380 GOSUB 5630
1390 GOSUB 5690
1400 GOSUB 5810
1410 GOSUB 5370
1420 GOSUB 5560
1430 GOSUB 5850
1440 GOSUB 5950
1450 GOSUB 6250
1460 IF A<>PP/2 THEN 1510
1470 IF B<>PP/2 THEN 1510
1480 PRINT :::"*** CORRECT!! ****"
1490 GOSUB 5590
1500 GOTO 1560
1510 PRINT :::"SORRY, THE REACTIONS ARE"
1520 PRINT :::"A =";PP/2
1530 PRINT "B =";PP/2
1540 GOSUB 5590
1550 GOTO 1570
1560 IF EX>2 THEN 1590
1570 EX=EX+1
1580 GOTO 1360
1590 GOSUB 6290
1600 IF KEY=49 THEN 1360
1610 IF KEY=51 THEN 5000
1620 I=16
1630 J=16
1640 GOSUB 5370
1650 GOSUB 5560
1660 CALL HCHAR(I+1,12,76)
1670 CALL HCHAR(I-5,16,80)
1680 INPUT "LENGTH OF BEAM = ":LL
1690 GOSUB 5310
1700 INPUT "LOAD P = ":PP
1710 GOTO 1410
1720 CALL CLEAR
1730 PRINT "GIVEN A BEAM OF LENGTH L"
1740 PRINT "SUPPORTED AT ENDS A AND B."
1750 PRINT "A CONCENTRATED FORCE OF"
1760 PRINT "P POUNDS IS APPLIED"
1770 PRINT "D FEET FROM A. IGNORE"
1780 PRINT "THE WEIGHT OF THE BEAM.":::
     :
1790 GOSUB 5810
1800 I=16
1810 J=12
1820 D=5
1830 GOSUB 5370
1840 GOSUB 5560
1850 CALL HCHAR(I-5,J,80)
1860 CALL HCHAR(I-1,9,68)
1870 CALL HCHAR(I+1,16,76)
1880 GOSUB 6040
1890 PRINT :"TAKING MOMENTS AT A"
1900 PRINT :"P*D - B*L = 0"
1910 PRINT TAB(7);"B*L = P*D"
1920 PRINT TAB(9);"B = P*D/L"
1930 PRINT :"NEXT SUM FORCES = 0"
1940 GOSUB 5590
1950 PRINT :"A + B - P = 0"
1960 PRINT "          A = P-B = P - P*D/L
     "
```

```
1970 GOSUB 5590
1980 RANDOMIZE
1990 EX=2
2000 PP=100*(INT(20*RND)+1)
2010 LL=INT(6*RND)+10
2020 D=INT(10*RND)+1
2030 GOSUB 5630
2040 GOSUB 5690
2050 GOSUB 5810
2060 GOSUB 5370
2070 J=INT(D/LL*21)+5
2080 GOSUB 5560
2090 GOSUB 5850
2100 GOSUB 5950
2110 DD$=STR$(D)
2120 FOR E=1 TO LEN(DD$)
2130 EE=J-5
2140 CALL HCHAR(I-1,EE+E,ASC(SEG$(DD$,E
,1)))
2150 NEXT E
2160 CALL HCHAR(I-1,EE+E,39)
2170 BB=PP*D/LL+.005
2180 BB=1E-2*(INT(BB*1E2))
2190 AA=PP-BB
2200 IF EX=2 THEN 2250
2210 GOSUB 6250
2220 IF AA<>A THEN 2240
2230 IF BB=B THEN 2300
2240 PRINT :"OUR ANSWERS DON'T AGREE.":
::
2250 GOSUB 6040
2260 GOSUB 6110
2270 GOSUB 6190
2280 EX=3
2290 GOTO 2000
2300 PRINT : : :"** YOU ARE CORRECT **"
2310 GOSUB 5590
2320 EX=EX+1
2330 IF EX<4 THEN 2000
2340 GOSUB 6290
2350 IF KEY=49 THEN 2000
2360 IF KEY=51 THEN 5000
2370 CALL CLEAR
2380 I=16
2390 J=12
2400 GOSUB 5370
2410 GOSUB 5560
2420 CALL HCHAR(I-5,J,80)
2430 CALL HCHAR(I-1,9,68)
2440 CALL HCHAR(I+1,16,76)
2450 INPUT "LENGTH OF BEAM = ":LL
2460 GOSUB 5310
2470 INPUT "LOAD P = ":PP
2480 INPUT "DISTANCE FROM A = ":D
2490 IF D>LL THEN 2510
2500 IF D>=0 THEN 2530
2510 PRINT :"SORRY, 0 <= D <= L":
:
2520 GOTO 2480
2530 GOTO 2060
2540 CALL CLEAR
2550 PRINT "GIVEN A SIMPLE BEAM"
2560 PRINT "SUPPORTED AT THE ENDS."
2570 PRINT "IT IS LENGTH L."
2580 PRINT "THERE IS A UNIFORM LOAD"
2590 PRINT "OF W POUNDS PER FOOT."
2600 GOSUB 5810
2610 CALL CLEAR
2620 PRINT "A UNIFORM LOAD CAN BE"
2630 PRINT "THOUGHT OF AS AN"
2640 PRINT "EQUIVALENT RESULTANT"
2650 PRINT "LOAD ACTING AT THE"
2660 PRINT "CENTROID OF THE LOADING":
:
:
:
:
:
:
:
2670 CALL HCHAR(21,5,112,7)
2680 CALL HCHAR(20,7,87)
2690 CALL HCHAR(20,8,47)
2700 CALL HCHAR(20,9,76)
2710 CALL HCHAR(22,8,76)
2720 CALL HCHAR(21,15,61)
2730 CALL HCHAR(22,19,115,7)
2740 I=22
2750 J=22
2760 GOSUB 5560
2770 CALL HCHAR(18,21,87)
2780 CALL HCHAR(21,23,76)
2790 CALL HCHAR(21,24,47)
2800 CALL HCHAR(21,25,50)
2810 GOSUB 5590
2820 GOSUB 6420
2830 PRINT "EQUIVALENT LOAD IS"
2840 PRINT "W*L ACTING AT CENTER."
2850 PRINT "SOLVING, A=B=W*L/2"
2860 GOSUB 5590
2870 EX=2
2880 RANDOMIZE
2890 WW=10*(INT(10*RND)+1)
2900 LL=INT(10*RND)+10
2910 GOSUB 5630
2920 GOSUB 5780
2930 GOSUB 5810
2940 GOSUB 6420
2950 GOSUB 5950
2960 GOSUB 6500
2970 GOSUB 6250
2980 AA=WW*LL/2
2990 BB=AA
3000 IF AA<>A THEN 3050
3010 IF BB<>B THEN 3050
3020 PRINT :"** CORRECT **"
3030 GOSUB 5590
3040 GOTO 3100
3050 PRINT :"A=B=W*L/2"
3060 PRINT "A=B=";AA;"POUNDS"
3070 GOSUB 5590
3080 EX=EX+1
3090 GOTO 2890
3100 I=18
3110 Y=16
3120 Z=5
3130 EX=EX+1
3140 GOSUB 6450
3150 PRINT "L = 16 FEET"
3160 PRINT "W = 80 LBS/FT"
3170 PRINT "ACTING 8 FT FROM A"
3180 PRINT "TO 12 FT FROM A":
:
3190 PRINT "EQUIVALENT FORCE IS"
3200 PRINT "80 LBS/FT * (12 FT - 8 FT)"
3210 PRINT "= 320 LBS"
3220 PRINT "APPLIED 10' FROM A."
3230 PRINT :"SUM MOMENTS ABOUT A."
3240 GOSUB 5590
3250 GOSUB 5370
3260 J=19
3270 GOSUB 5560
3280 LL=16
3290 PP=320
3300 GOSUB 5850
3310 GOSUB 5950
3320 CALL HCHAR(I-1,10,49)
3330 CALL HCHAR(I-1,11,48)
3340 CALL HCHAR(I-1,12,39)
3350 PRINT "320# * 10' - B*16' = 0"
3360 PRINT TAB(5);"B=3200/16 LBS = 200
LBS":
:
3370 PRINT "NOW SUM FORCES"
3380 GOSUB 5590
3390 PRINT :"320# - 200# - A = 0"
3400 PRINT "A = 120 LBS"
3410 GOSUB 5590
3420 EX=EX+1
3430 RANDOMIZE
3440 GOSUB 6660
3450 GOSUB 6610
3460 GOSUB 6730
3470 GOSUB 5590
3480 GOSUB 5370
3490 GOSUB 6370
3500 GOSUB 6500
```

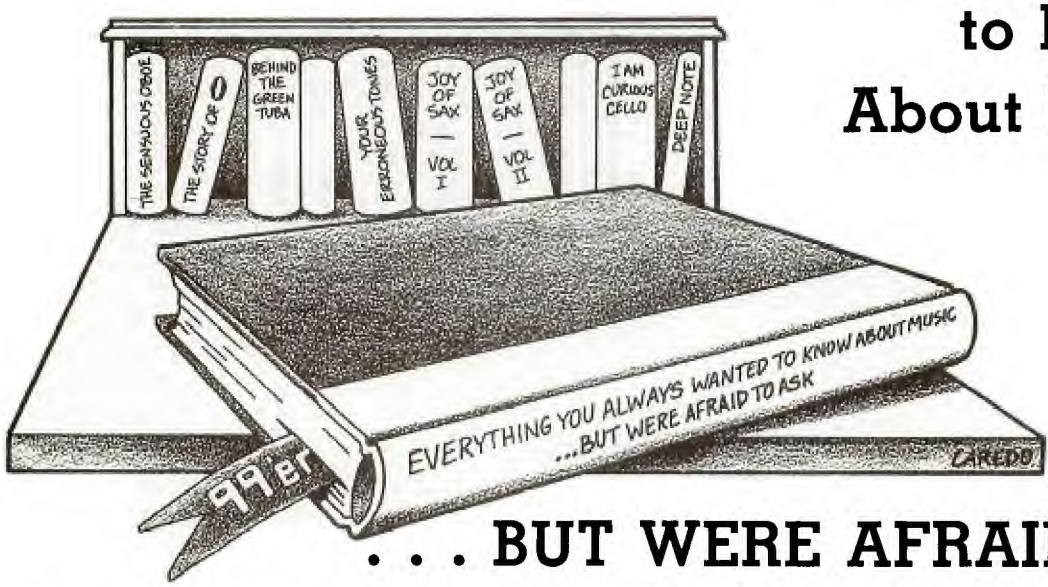
```
3510 GOSUB 5950
3520 GOSUB 6250
3530 BB=LOAD*(D2/2+L1)/LL+.005
3540 BB=1E-2*(INT(BB*1E2))
3550 AA=LOAD-BB
3560 IF AA<>A THEN 3610
3570 IF BB<>B THEN 3610
3580 PRINT :"** CORRECT **"
3590 GOSUB 5590
3600 GOTO 3660
3610 PRINT :"SORRY.  IT IS "
3620 PRINT "A = ";AA
3630 PRINT "B = ";BB:
3640 GOSUB 5590
3650 GOTO 3420
3660 GOSUB 6290
3670 IF KEY=49 THEN 3420
3680 IF KEY=51 THEN 5000
3690 CALL CLEAR
3700 I=16
3710 Y=16
3720 Z=5
3730 GOSUB 6450
3740 INPUT "LENGTH OF BEAM IN FT = ":LL
3750 GOSUB 5310
3760 INPUT "LOADING W LB/FT = ":WW
3770 IF WW<>0 THEN 3800
3780 PRINT "IF W=0,  A=B=0"
3790 GOTO 3760
3800 INPUT "ACTING AT DISTANCE FROM A "
:L1
3810 IF L1<0 THEN 3830
3820 IF L1<LL THEN 3850
3830 PRINT "SORRY,  0 <= L1 < LL"
3840 GOTO 3800
3850 INPUT "TO - DISTANCE FROM A ":L2
3860 IF L2<=L1 THEN 3880
3870 IF L2<=LL THEN 3900
3880 PRINT "SORRY,  L1 < L2 <= L"
3890 GOTO 3850
3900 GOSUB 6730
3910 GOTO 3480
3920 CALL CLEAR
3930 PRINT TAB(4);"COMBINATION LOADS":
:
3940 PRINT :"TO SOLVE THIS TYPE PROBLEM
:"
3950 PRINT :"DRAW AND LABEL THE BEAM."
3960 PRINT :"SUM MOMENTS ABOUT A OR B."
3970 PRINT :  :"SUM FORCES":  :
3980 GOSUB 5590
3990 CALL CLEAR
4000 RANDOMIZE
4010 GOSUB 6660
4020 GOSUB 6730
4030 PP=100*(INT(15*RND))
4040 D=INT(LL*RND)
4050 GOSUB 5650
4060 IF PP=0 THEN 4080
4070 GOSUB 5710
4080 IF WW=0 THEN 4130
4090 GOSUB 5780
4100 IF L1=LL THEN 4120
4110 GOSUB 6630
4120 GOSUB 5810
4130 I=16
4140 J=INT(D/LL*22)+5
4150 GOSUB 5370
4160 GOSUB 5950
4170 IF WW=0 THEN 4200
4180 GOSUB 6370
4190 GOSUB 6500
4200 IF PP=0 THEN 4230
4210 GOSUB 5560
4220 GOSUB 5850
4230 GOSUB 6250
4240 BB=(PP*D+LOAD*(D2/2+L1))/LL+.005
4250 BB=1E-2*(INT(BB*1E2))
4260 AA=PP+LOAD-BB
4270 IF ABS(AA-A)>.01 THEN 4340
4280 IF ABS(BB-B)>.01 THEN 4340
4290 PRINT :  :"** CORRECT **"
4300 GOSUB 5590
4310 GOSUB 6290
4320 IF KEY=49 THEN 3990
4330 IF KEY=51 THEN 5000 ELSE 4390
4340 PRINT :  :"SORRY,  THE ANSWER I GET
IS"
4350 PRINT :"A = ";AA
4360 PRINT "B = ";BB
4370 GOSUB 5590
4380 GOTO 3990
4390 CALL CLEAR
4400 PRINT "YOU MAY ENTER YOUR OWN"
4410 PRINT :"PROBLEMS.  JUST ENTER"
4420 PRINT :"ALL VARIABLES AS "
4430 PRINT :"SHOWN ON THE DIAGRAM."
4440 PRINT :  :"I WILL GIVE THE ANSWERS.
":  :  :  :  :
4450 GOSUB 5590
4460 I=16
4470 J=12
4480 L1=0
4490 L2=0
4500 D2=0
4510 LOAD=0
4520 GOSUB 5370
4530 GOSUB 5560
4540 CALL HCHAR(I-5,J,80)
4550 Y=16
4560 Z=5
4570 GOSUB 6370
4580 CALL HCHAR(I-3,Y+2,87)
4590 CALL HCHAR(I+2,13,76)
4600 CALL HCHAR(I+1,Y,76)
4610 CALL HCHAR(I+1,Y+1,49)
4620 CALL HCHAR(I+1,Y+Z-1,76)
4630 CALL HCHAR(I+1,Y+Z,50)
4640 CALL HCHAR(I-1,8,68)
4650 INPUT "LENGTH OF BEAM L = ":LL
4660 GOSUB 5310
4670 INPUT "FORCE P = ":PP
4680 IF PP=0 THEN 4740
4690 INPUT "DISTANCE D = ":D
4700 IF D>LL THEN 4720
4710 IF D>=0 THEN 4740
4720 PRINT :"SORRY,  0 <= D <= L":
4730 GOTO 4690
4740 INPUT "LOADING W = ":WW
4750 IF WW=0 THEN 4880
4760 INPUT "DISTANCE FROM A,L1 = ":L1
4770 IF L1<0 THEN 4790
4780 IF L1<LL THEN 4810
4790 PRINT "SORRY,  0 <= L1 <L"
4800 GOTO 4760
4810 INPUT "DISTANCE FROM A,L2 = ":L2
4820 IF L2<=L1 THEN 4840
4830 IF L2<=LL THEN 4860
4840 PRINT "SORRY,  L1 < L2 <= LL"
4850 GOTO 4810
4860 D2=L2-L1
4870 LOAD=WW*D2
4880 BB=(PP*D+LOAD*(D2/2+L1))/LL+.005
4890 BB=1E-2*(INT(BB*1E2))
4900 AA=PP+LOAD-BB
4910 PRINT :  :"A = ";AA;"POUNDS"
4920 PRINT "B = ";BB;"POUNDS"
4930 GOSUB 5590
4940 PRINT :"ANOTHER PROBLEM?(Y/N) "
4950 CALL KEY(0,KEY,ST)
4960 IF KEY=89 THEN 4480
4970 IF KEY=78 THEN 5000
4980 GOTO 4950
4990 END
5000 CALL CLEAR
5010 CALL SCREEN(5)
5020 FOR E=1 TO 8
5030 CALL COLOR(E,12,12)
```

```
5040 NEXT E
5050 PRINT :::::::::
5060 PRINT "SELECT":::
5070 PRINT :"1 CONCENTRATED LOAD, CENTE
R"
5080 PRINT :"2 CONCENTRATED LOAD ANYWHE
RE"
5090 PRINT :"3 UNIFORM LOADS"
5100 PRINT :"4 COMBINATION LOADS"
5110 PRINT :"5 PROBLEMS ONLY"
5120 PRINT :"6 YOUR OWN PROBLEMS"
5130 PRINT :"7 END PROGRAM"::::
5140 CALL VCHAR(1,2,32,24)
5150 CALL VCHAR(1,31,32,24)
5160 CALL VCHAR(1,1,32,24)
5170 CALL VCHAR(1,32,32,24)
5180 FOR E=1 TO 8
5190 CALL COLOR(E,2,12)
5200 NEXT E
5210 CALL KEY(0,KEY,ST)
5220 CHOICE=KEY-48
5230 IF CHOICE<1 THEN 5210
5240 IF CHOICE>7 THEN 5210
5250 CALL CLEAR
5260 CALL SCREEN(8)
5270 FOR E=1 TO 8
5280 CALL COLOR(E,2,1)
5290 NEXT E
5300 ON CHOICE GOTO 700,700,2540,3920,3
990,4390,4990
5310 IF LL>=1 THEN 5360
5320 PRINT "HEY, WHAT KIND OF BEAM"
5330 PRINT "HAS A LENGTH LIKE THAT?!!"
5340 INPUT "TRY AGAIN; L = ":LL
5350 GOTO 5310
5360 RETURN
5370 CALL CLEAR
5380 CALL HCHAR(I,5,120)
5390 CALL HCHAR(I,6,121,21)
5400 CALL HCHAR(I,27,122)
5410 CALL HCHAR(I+1,5,99)
5420 CALL HCHAR(I+1,27,100)
5430 FOR K=4 TO 26 STEP 22
5440 CALL HCHAR(I+2,K,101)
5450 CALL HCHAR(I+2,K+1,102)
5460 CALL HCHAR(I+2,K+2,103)
5470 CALL VCHAR(I+3,K+1,104)
5480 CALL VCHAR(I+4,K+1,105,2)
5490 NEXT K
5500 CALL HCHAR(I+1,4,65)
5510 CALL HCHAR(I+1,28,66)
5520 RETURN
5530 FOR DELAY=1 TO 1000
5540 NEXT DELAY
5550 RETURN
5560 CALL VCHAR(I-4,J,105,3)
5570 CALL VCHAR(I-1,J,106)
5580 RETURN
5590 PRINT :"PRESS ENTER TO CONTINUE"
5600 CALL KEY(0,KEY,ST)
5610 IF KEY<>13 THEN 5600
5620 RETURN
5630 CALL CLEAR
5640 PRINT "PROBLEM":::
5650 PRINT "GIVEN A SIMPLE BEAM"
5660 PRINT "SUPPORTED AT THE ENDS."
5670 PRINT "IT IS";LL;"FEET LONG."
5680 RETURN
5690 IF CHOICE>2 THEN 5710
5700 PRINT :"IGNORE WEIGHT OF THE BEAM.
"
5710 PRINT "A CONCENTRATED LOAD OF"
5720 PRINT PP;"POUNDS IS"
5730 IF CHOICE=1 THEN 5760
5740 PRINT D;"FEET FROM END A."
5750 RETURN
5760 PRINT "AT THE CENTER OF THE BEAM."
5770 RETURN
5780 PRINT :"THERE IS A UNIFORM LOAD"
5790 PRINT "OF";WW;"POUNDS/FOOT"
5800 RETURN
5810 PRINT :"FIND THE REACTION FORCES."
::
5820 PRINT "DRAW AND LABEL THE PROBLEM.
":::
5830 GOSUB 5590
5840 RETURN
5850 LB$=STR$(PP)
5860 FOR II=1 TO LEN(LB$)
5870 JJ=II+J-4
5880 CALL HCHAR(I-5,JJ,ASC(SEG$(LB$,II,
1)))
5890 NEXT II
5900 CALL HCHAR(I-5,JJ+1,32)
5910 CALL HCHAR(I-5,JJ+2,76)
5920 CALL HCHAR(I-5,JJ+3,66)
5930 CALL HCHAR(I-5,JJ+4,83)
5940 RETURN
5950 FT$=STR$(LL)
5960 FOR II=1 TO LEN(FT$)
5970 JJ=12+II
5980 CALL HCHAR(I+1,JJ,ASC(SEG$(FT$,II,
1)))
5990 NEXT II
6000 CALL HCHAR(I+1,JJ+2,70)
6010 CALL HCHAR(I+1,JJ+3,69,2)
6020 CALL HCHAR(I+1,JJ+5,84)
6030 RETURN
6040 CALL HCHAR(23,3,98)
6050 CALL HCHAR(23,4,77)
6060 CALL HCHAR(23,6,61)
6070 CALL HCHAR(23,8,48)
6080 PRINT "WRITE THE EQUATION"
6090 GOSUB 5590
6100 RETURN
6110 PRINT :"TAKING MOMENTS AT A."
6120 PRINT :"P*D - B*L = 0"
6130 PRINT "B = P * D/L"
6140 PRINT "B =";PP;"*";D;"/";L
L
6150 PRINT "B =";BB;"POUNDS"
6160 PRINT :"NOW FIND A."
6170 GOSUB 5590
6180 RETURN
6190 PRINT :"SUM OF FORCES = 0"
6200 PRINT "P-A-B = 0"
6210 PRINT "A = P-B =";PP;"-";BB
6220 PRINT "A =";AA;"POUNDS"
6230 GOSUB 5590
6240 RETURN
6250 PRINT :"WHAT ARE A AND B IN POUNDS
?"
6260 INPUT "A = ":A
6270 INPUT "B = ":B
6280 RETURN
6290 PRINT ::"DO YOU WANT MORE PROBLEM
S?"
6300 PRINT :"1 YES, SAME KIND"
6310 PRINT "2 YES, MY OWN PROBLEMS"
6320 PRINT "3 NO, DO SOMETHING ELSE"
6330 CALL KEY(0,KEY,ST)
6340 IF KEY<49 THEN 6330
6350 IF KEY>51 THEN 6330
6360 RETURN
6370 CALL HCHAR(I-1,Y,112,Z)
6380 CALL HCHAR(I-2,Y,112,Z)
6390 CALL VCHAR(I-2,Y-1,114,2)
6400 CALL VCHAR(I-2,Y+Z,113,2)
6410 RETURN
6420 I=16
6430 Y=6
6440 Z=21
6450 GOSUB 5370
6460 GOSUB 6370
6470 CALL HCHAR(I-3,16,87)
6480 CALL HCHAR(I+1,16,76)
```

```
6490 RETURN
6500 X=INT(Y+Z/2-3)
6510 UL$=STR$(WW)
6520 FOR E=1 TO LEN(UL$)
6530 CALL HCHAR(I-3,X+E-1,ASC(SEG$(UL$,
     E,1)))
6540 NEXT E
6550 CALL HCHAR(I-3,X+E,76)
6560 CALL HCHAR(I-3,X+E+1,66)
6570 CALL HCHAR(I-3,X+E+2,47)
6580 CALL HCHAR(I-3,X+E+3,70)
6590 CALL HCHAR(I-3,X+E+4,84)
6600 RETURN
6610 GOSUB 5630
6620 GOSUB 5780
6630 PRINT "ACTING FROM";L1;"FEET FROM
     A"
```

```
6640 PRINT "TO";L2;"FEET FROM A"
6650 RETURN
6660 LL=INT(8*RND)+12
6670 WW=10*(INT(4*RND)+5)
6680 LIM1=INT(3*LL/4)
6690 L1=INT(LIM1*RND)
6700 LIM2=LL-L1
6710 L2=INT(LIM2*RND)+L1+1
6720 RETURN
6730 D2=L2-L1
6740 LOAD=WW*D2
6750 Y=INT(L1/LL*22)+6
6760 Z=INT(D2/LL*22)-1
6770 RETURN
```

Almost Everything You Ever Wanted to Know About Music . . .



. . . BUT WERE AFRAID TO ASK

Is music terminology Greek to you? Do you feel deficient in certain areas of your musical ability? How are your listening skills? If you enjoy music and want to test and improve your abilities, TI's *Music Skills Trainer* can be a valuable tool. This program provides practice in aural recognition of pitches, intervals, and chords, and develops your ability to remember musical phrases. You can control the complexity of each drill by selecting various options including note range, use of sharps (♯) and flats (♭), types of chords and intervals, and the playing of random music between examples.

Since the program is designed to provide drill and does not teach the underlying concepts involved, this article will first cover relevant aspects of music theory. We'll then follow it up with a review of *Music Skills Trainer*.

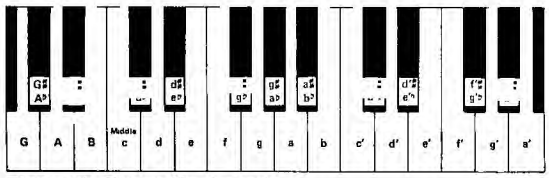


Figure 1. Piano Keyboard

The Scale

The fundamental concept involved is that of the *scale*—an ordered group of tones within an octave. The **C** Major scale, with which almost everyone is familiar, provides the standard pattern for every major scale (Do-Re-Mi-Fa-Sol-La-Ti-Do). This pattern originated with the Greeks and is based upon the *tetrachord*. A tetrachord can be thought of as half a scale; it consists of four tones arranged so that they contain two whole steps followed by a half step. Refer to the diagram of a piano keyboard in Figure 1. Starting at middle **c**, each progression up the keyboard represents a half step or semitone. For example, all the following represent half steps: **c-c ♯,c ♯-d, d-d ♯, d ♯-e, e-f**, etc. The first tetrachord for the **C** Major scale consists of the following two whole steps: **c-d** and **d-e** followed by the natural half step **e-f**. The second tetrachord begins with **g** and again consists of two whole steps followed by a half step, ending with **c'** (an octave above middle **c**). This tetrachord pattern $(1 + 1 + ½)$ was referred to as a *diatonic scale*.

In order to accommodate Oriental and other music, Greek theorists modified the two middle tones of the diatonic tetrachord in several ways. One of these, called the *chromatic tetrachord*, consisted of the pattern $1½ + ½ + ½$ (e.g., **c, d ♯, e, f,**). Various combinations of these two tetrachords necessitate the division of an octave into the familiar twelve equally spaced intervals referred to as the chromatic scale: **c, c ♯, d, d ♯, e, f, f ♯, g, g ♯, a, a ♯, b, c'**.

Pitch refers to the location of one of these tones in a scale, and is defined by a regular frequency of vibrations. In the United States the standard assignment for **a** above middle **c** is 440 vibrations per second. It happens that a pure octave differs from any reference pitch by a factor of exactly 2, so that **a** two octaves above middle **c** = 880 and **A** below middle **c** = 220.

Although knowledge of frequencies is not required for use of the *Music Skills Trainer*, you may be interested to know how frequencies are assigned to other scale positions. Because each octave is divided into twelve equally-spaced intervals, the factor $2^{\frac{1}{12}}$ is used to define the relative frequencies of successive tones. For example,

$$\text{If } \mathbf{a} = 440;$$
$$\mathbf{a}\sharp = 440 \times 2^{\frac{1}{12}},$$
$$\mathbf{b} = \mathbf{a}\sharp \times 2^{\frac{1}{12}} = \mathbf{a} \times 2^{\frac{1}{12}} \times 2^{\frac{1}{12}} = \mathbf{a} \times (2^{\frac{1}{12}})^2.$$

Given any reference frequency, f_0, then the relative pitch of any other scale position, f, can be calculated by counting the number of half steps to that position, N, and using the formula:

$$f = f_0(2^{\frac{1}{12}})^N.$$

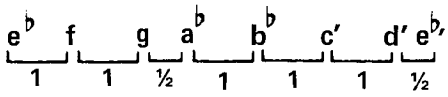
The following program calculates and plays a chromatic scale beginning with middle **c** (262).

```
100 REM ***********
110 REM * MUSIC 1 *
120 REM ***********
130 REM
```

```
140 REM
150 REM
160 REM
170 F0=262
180 FOR N=0 TO 12
190 F=F0*(2^(1/12))^N
200 CALL SOUND(-600,F,0)
210 NEXT N
220 STOP
```

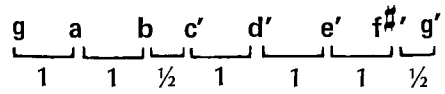
Scales in Various Keys

Now let us return to the diatonic (major) scale. A major scale can have a starting or *root* note of any of the twelve chromatic pitches. As in the case discussed above, a major scale is constructed, starting from the root, with two diatonic tetrachords (1 + 1 + ½) separated by a whole step. A more convenient way to construct a major scale is simply to remember that half steps occur between the third and fourth and the seventh and eighth tones. Referring to Figure 1, a major scale with $e^{\flat}$ as the root would be constructed using the following steps:



Steps

This scale is referred to as an $E^{\flat}$ Major scale, or a scale in the Key of $E^{\flat}$, since $e^{\flat}$ is the root. Similarly, a major scale in the key of **G** is constructed as follows:



Steps

Steps

While there are twelve such different diatonic scales, they all sound the same because they are based on the same pattern of diatonic steps. The following program plays these scales beginning with **C** Major.

```
100 REM  **********
110 REM  *  MUSIC  2  *
120 REM  **********
130 REM
140 REM
150 REM
160 REM
170 F0=262
180 FOR I=0 TO 11
190 F=F0*(2^(1/12))^I
200 FOR I=1 TO 8
210 N=2
220 IF I=4 THEN 240
230 IF I=8 THEN 240 ELSE 250
240 N=1
250 F=F*(2^(1/12))^N
260 CALL SOUND(-600,F,0)
270 NEXT J
280 NEXT I
290 STOP
```

Intervals

An *interval* is the difference in pitch between two notes. Interval names indicate the number of included tones of the major scale. Starting with middle **c** in Figure 1, the basic interval names are as follows: **c-c**, unison (prime); **c-d**, second; **c-e**, third; **c-f**, fourth; **c-g**, fifth; **c-a**, sixth; **c-b**, seventh; and **c-c'**, octave. **c-f** is a fourth because it includes the following diatonic tones of the **C** Major scale: **c**, **d**, **e**, and **f**. Similarly in the $E^{\flat}$ Major scale, a fourth is $e^{\flat}$-$a^{\flat}$, and in the **G** Major scale a fourth is **g-c'**. However, as in the case of scales, an interval in one key sounds like that interval in another.

Four of the eight intervals can exist in one of four forms. If the upper note of the interval lies within the major scale of the lower or root note, the interval may be classified as *major*. If the upper note is lowered a half step, however, the interval then becomes *minor*. For example, **c-e** is a major third and **c-e**$^{\flat}$ a minor third. This rule applies to four intervals; the second,third, sixth, and seventh. The remaining intervals—fourth,fifth, and octave—are classified as *perfect*: They do not exist in major and minor forms. The following program plays all of the intervals above in the **C** Major scale, i.e., with middle **c** as the lower or root note.

The remaining two categories of intervals—*augmented* and *diminished*—are not used in the *TI Music Skills Trainer* and so will not be discussed in detail. They are formed as follows: augmented—a major or perfect interval is made one half step larger; diminished—a minor or perfect interval is made one half step smaller.

Finally, intervals may be classified according to which note is played first. If the lower note is played first, the interval is said to be ascending (**c-e**), and if the upper note is played first, it is descending (**e-c**).

Chords

A chord is several notes played simultaneously, usually three or more. When a chord consists of three tones it is called a *triad*. Given any major scale, four triads can be formed from the starting note (root) of that scale: major, minor, augmented, and diminished. A major triad consists of the root, the third, and the fifth. For example, in a **C** Major scale, starting with the root **c**, the third is **c-e**, and the fifth is **c-g**. The major chord is then **c-e-g**. Similarly, in the $E^{\flat}$ Major scale, given the root $e^{\flat}$, the third **g**, and the fifth $b^{\flat}$, the major chord is $e^{\flat}$-**g**-$b^{\flat}$.

The major chord is changed to a minor chord by lowering the second note (i.e., the third) one half step. For example, the **C** Major chord **c-e-g** becomes the **C** minor chord **c**-$e^{\flat}$-**g** and the $E^{\flat}$ Major chord becomes the $E^{\flat}$ minor chord $e^{\flat}$-$g^{\flat}$-$b^{\flat}$.

A minor chord can further be changed to a diminished chord by lowering the third note (i.e., the fifth) one half step. For example, the **C** minor chord **c**-$e^{\flat}$-**g** becomes the **c** diminished chord **c**-$e^{\flat}$-$g^{\flat}$ and the $E^{\flat}$ minor chord $e^{\flat}$-$g^{\flat}$-$b^{\flat}$ becomes the $E^{\flat}$ diminished chord $e^{\flat}$-$g^{\flat}$-$b^{\flat\flat}$. ($b^{\flat\flat}$ is called "**b** double flat" and is the same note as **a**.)

The augmented chord is formed by raising the third note of the major chord (i.e., the fifth) one half step. For example, the **C** Major chord **c-e-g** becomes the **C** augmented chord **c-e-g** ♯ and the $E^{\flat}$ Major chord becomes the $E^{\flat}$ augmented chord $e^{\flat}$-**g**-**b**.

As in the case of scales and intervals, chords with the same name sound alike. All major chords sound alike; all minor chords sound alike, etc.

If the lowest note of the chord is the root, the chord is said to be in *root position*. All four types of triads (chords), however, can be played in inverted form. For example, the **C** Major chord **c-e-g** may be altered from its root position form to one of the following inversions by making the lowest note either the third or the fifth: **e-g-c'** and **g-c'-e'**. Similar-

ly, the inverted forms for the E♭ minor—which in root position is written or played e♭-g♭-b♭—are g♭-b♭-e♭' and b♭-e♭'-g♭'.

Chords of more than three notes can be formed, and there are several different varieties. One of them, the seventh, is used in the *Music Skills Trainer*. The seventh chord contains the root, third, fifth, and the seventh lowered by a half step. For example, a seventh in the key of **C** Major is **c-e-g** and **b** lowered by a half step or b♭. Similarly, in the key of e♭ the seventh chord is e♭-g-b♭-d♭ (**d** lowered by a half step).

While the seventh chord contains four notes, the TI-99/4A can play only three notes simultaneously; therefore, following traditional rules of harmony the fifth of the chord (third note) may be omitted to give a seventh in the form of **c-e-b♭**. As in the case of triads, the seventh may appear in inverted forms.

TI *Music Skills Trainer*

The *Music Skills Trainer* from Texas Instruments is a program written in TI BASIC (it will also run in Extended BASIC without modification). The program is available on cassette or diskette.

Four types of drill are provided: Pitch Guess, Interval Recognition, Chord Recognition, and Phrase Recall. The user selects the type of drill desired from a menu.

Pitch Guess

In this drill, you try to identify the pitch of a single note. While it might seem that this would require perfect pitch, you will find after several examples that you have "tuned in" and are able to identify pitches by relating each new one to the one that has preceded. The difficulty of this exercise can be controlled by specifying the starting note and range size in half steps (up to two octaves). In addition, you can choose to have notes selected from either the **C** Major diatonic or chromatic scales by answering "No" or "Yes" to the option of including sharps and flats. TI has included yet another means of increasing the level of difficulty—Random Music. If chosen, random music is played between examples, making it more difficult to remember the preceding note. The program provides up to ten examples and keeps score: 10 points for each correct answer.

We recommend that when first using this drill, you use **c** as the starting note, a range size of 13 (one octave), no sharps and flats, and no random music. After a little practice, it shouldn't be that difficult to identify notes.

Interval Recognition

This drill helps to develop your ability to recognize intervals. There are three levels, each of which adds more intervals to those included in the drill. For instance, if you choose Level 1, the examples are composed of major thirds, fourths, and fifths. Level 2 adds half steps, whole steps, and minor thirds. and Level 3 sixths, sevenths, and octaves. You can choose to have the intervals presented in ascending or descending order. For an added difficulty, you can also choose to have the lower note be random; it is otherwise **c** each time. You can also choose to have random music play between exercises. Up to ten examples are provided, and you receive 10 points for each correct answer.

Chord Recognition

This drill provides practice in recognizing chords. Again there are three levels, with Level 1 consisting of major and minor chords, Level 2 adding seventh and diminished, and Level 3 adding augmented. If you choose the Random Bass option, the root can be any note; otherwise it is a **c**. If you choose the Random Inversions option, inverted chords will be played; otherwise, a root-position chord is always played. If you choose the Chord Only option, the three notes will be played simultaneously. If you don't choose it, the notes comprising the chord are first played individually and then together. As in the previous drills, you can select the Random Music option. You receive 10 points for each, up to 10 problems.

Phrase Recall

This drill develops your ability to remember a sequence of as many as nine random notes. A blank keyboard overlay, provided with the program, is used to label the keys with their corresponding pitch, covering two octaves much like the layout of a piano keyboard. You can select the starting note and range size, and determine whether sharps and flats are to be included in the examples. You can also specify the number of notes which constitute the phrase (1-9). After a phrase is played, you respond by entering notes from the keyboard as if it were a piano. As you play the notes, you hear them and they are displayed as well; if you make a mistake, you can use SHIFT T to start over again without penalty. When you have entered the notes that you think correctly represent the phrase, you press ENTER. The correct notes are then displayed below your response, and you are awarded points based on the number of correct notes and the number of notes included in the phrase. Up to ten examples are given with a possible total score of up to 100 points. As in the previous drills, the Random Music option can be chosen to make this drill even more difficult.

We feel that TI's *Music Skills Trainer* will be useful even for experienced musicians who want to keep their auditory skills sharp. We would also recommend it for novices interested in further developing their knowledge and abilities in areas of music theory covered by the program.

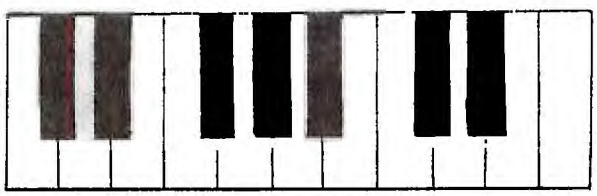
Let's Learn Notes

Let's Learn Notes was designed for beginning music students. A piano or organ teacher can use the program during a lesson to give the student a different approach to learning musical notes, or a student can run the program before or after the regular lesson. Students can also use the program at home for additional practice in learning musical notes. Even preschool children can begin learning the notes with this program.

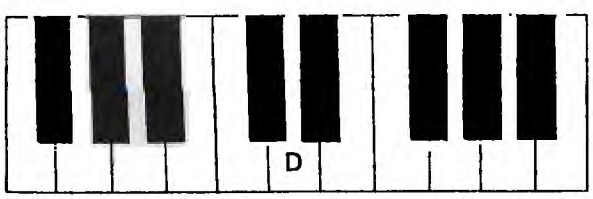
The program is written in TI BASIC and uses color graphics to draw piano keyboards, musical staves, and notes. In addition, the program generates musical tones.

This program provides three options: Keyboard Learning, Treble Clef Learning and Bass Clef Learning. Each option asks for ten responses. An incorrect response is recognized by a slight non-musical noise; the correct response must be entered before the program will continue.

Keyboard Learning randomly selects and displays one of two piano keyboards (starting at the left with either two black keys or three black keys). It then randomly selects one of the 11 displayed piano keys and flashes a question mark on the key. The student responds by pressing the letter on the computer keyboard that corresponds to the letter name of the piano key shown. If the response is correct, the corresponding musical tone is played and the letter name is printed on the piano key. The program randomly chooses Keyboard 1 or Keyboard 2 for each question. If the keyboard chosen is the same as for the previous question, the keyboard is not redrawn.

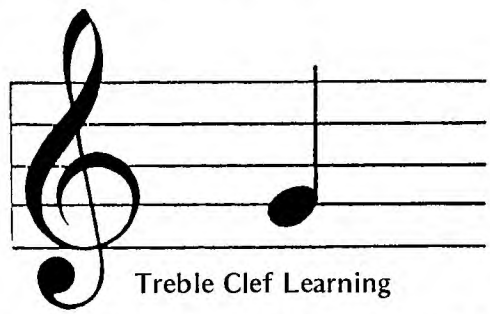


Keyboard 1



Keyboard 2

Treble Clef Learning displays a staff and treble clef. A note is selected randomly from Middle **C** to high **F** (top line of the staff) and displayed as a red quarter note. The student presses the letter on the computer keyboard that corresponds to the letter name of the note. If the response is correct, the corresponding musical tone is played and the letter name is printed on the note.



Treble Clef Learning

Bass Clef Learning displays a staff and bass clef. A note is selected randomly from low **G** (bottom line of the staff) to Middle **C** and displayed as a red quarter note. The student presses the letter on the computer keyboard that corresponds to the letter name of the note. If the response is correct, a five-note scale is played and the letter name is printed on the note.

Bass Clef Learning

This program is very easy to use and "student-friendly"—even for the youngest piano learners. A student can select the three learning options either at the beginning of the program or after each option has finished, simply by pressing 1, 2, or 3 on the computer keyboard. If a number greater than 3 is pressed, the program ends.

This program makes repetitious drill much more fun for the piano student and much less boring for the teacher. TI's color graphics and sound in this program greatly enhance the student's motivation to learn the letter names of piano keys and notes.

EXPLANATION OF THE PROGRAM
Let's Learn Notes

Line Nos.	
150	T=1500 for the CALL SOUND(T,---,-) statements.
170-450	Defines colors and characters for the title screen.
460-1220	Displays the characters for musical notes and a treble clef for the title screen. Musical tones of the **C** Major scale and arpeggio are played while the title screen is displayed.
1230	Asks which option the student wants and branches to that option.
1240-1340	Option 1, Keyboard. Defines color and characters for drawing the keyboard.
1360	COUNT set to zero and incremented for each question. There are 10 questions in each option.
1370	Keyboard number is randomly chosen, 1 or 2.
1380-1440	Prints "NAME THE KEY".
1450-1550	Draws the white keys.
1570-1730	Draws the black keys for one of the two piano keyboards.
1740-1750	Chooses one of the 11 keys randomly.
1760-1810	Blinks a red question mark on the key.
1820-1830	Reads the student's response.
1840-2680	Tests the response. If it is incorrect, there is a nonmusical sound and another response is required. If it is correct, the corresponding musical tone is played and the letter name of the key is displayed on the key.
2690-2710	Delays, then erases the letter name.
2720-2730	Increments COUNT and determines if there have been 10 questions.
2740-2750	Chooses keyboard pattern randomly. If it is the same as the previous question, only a new key is chosen; if it is different, a new keyboard is drawn before the key is chosen.
2760	Treble Clef and Bass Clef option.
2770-2800	Resets colors for this screen.
2810-3270	Defines special characters for staff, treble clef, and note.
3280	Draws staff.
3290-3340	Prints "NAME THE NOTE".
3360-3450	Treble Clef option. Draws the treble clef.
3460	Sets COUNT for number of problems.
3470-3530	Chooses note and draws it.
3540-3550	Reads the student's response.
3560-4010	Tests the response. If it is incorrect, there is a nonmusical sound and another response is required. If it is correct, the corresponding musical tone is played and the letter name is displayed on the note.
4020-4030	After a delay, erases the note and chooses a new note. If there have been 10 notes, the options are listed again.
4040-4160	Bass Clef option. Defines special characters for the bass clef.
4170-4230	Prints bass clef.
4240	Sets COUNT = 0 for the number of problems.
4250-4310	Chooses one of 11 notes randomly and draws it.
4320-4330	Reads the student's response.
4340-4900	Tests the response. If it is incorrect, there is a nonmusical sound and another response is required. If it is correct, a five-note scale is played starting at frequency J, and the letter name is displayed on the note.
4910-4920	After a delay, erases the note and chooses a new note. If there have been 10 notes, the options are listed again.
4930-4970	Subroutine for playing the 5-note scale.
4980-5060	Subroutine for drawing the staff.
5070-5210	Subroutine for drawing the note.
5130-5210	Draws the stem of the note up or down from the note, depending on where the note is.
5220-5390	Subroutine for procedure after each note.
5230-5240	Delays
5250-5260	Increments and tests COUNT.
5260-5390	If COUNT < 10, erases the note and returns.
5400-5430	If COUNT = 10, prints menu screen of options.
5540-5490	Branches to Option 1, 2, or 3.
5500	If 4 is pressed, the program ends.

```
100 REM *********************
110 REM *LET'S LEARN NOTES*
120 REM *********************
130 REM
140 REM
150 T=1500
160 CALL CLEAR
170 CALL COLOR(11,13,1)
180 CALL COLOR(12,13,1)
190 CALL COLOR(13,14,1)
200 CALL COLOR(14,5,1)
210 CALL COLOR(15,5,1)
220 CALL CHAR(112,"0000002070505050")
230 CALL CHAR(113,"0505050303020202")
240 CALL CHAR(114,"050D09103020604")
250 CALL CHAR(115,"0000000808080809CE3")
260 CALL CHAR(116,"40C1C2C286868C2")
270 CALL CHAR(117,"C0404040202020202")
280 CALL CHAR(118,"8080404040404040")
290 CALL CHAR(119,"C140402010007")
300 CALL CHAR(120,"2020101111210")
310 CALL CHAR(121,"8080807")
320 CALL CHAR(122,"10999010")
330 CALL SOUND(T,262,2)
340 CALL CHAR(136,"0000030F1F1F3F3F")
350 CALL CHAR(137,"0202E2FAFAFEFEFE")
360 CALL CHAR(138,"1F1F0F03")
370 CALL CHAR(139,"FCFCF8E0")
380 CALL CHAR(140,"0202020202020202")
390 CALL CHAR(141,"0000000030302020")
```

```
400 CALL CHAR(142,"0000000000080402020")
410 CALL CHAR(143,"2020101010080808")
420 CALL CHAR(144,"000111EE")
430 CALL CHAR(145,"0FF0")
440 CALL CHAR(146,"00000000000031CE0")
450 CALL CHAR(147,"000000003EC20202")
460 CALL SOUND(T,294,2)
470 DATA 76,69,84,39,83,32,76,69,65,82
    ,78,32
480 RESTORE 470
490 FOR Y=12 TO 23
500 READ L
510 CALL HCHAR(6,Y,L)
520 NEXT Y
530 CALL SOUND(T,330,2)
540 DATA 78,79,84,69,83,32
550 RESTORE 540
560 FOR Y=13 TO 23 STEP 2
570 READ L
580 CALL HCHAR(10,Y,L)
590 NEXT Y
600 CALL SOUND(T,349,2)
610 DATA 8,4,138,8,5,139,7,4,136,7,5,1
    37,6,5,140,5,5,140,4,5,141,4,6,144
    ,4,7,145
620 RESTORE 610
630 FOR I=1 TO 9
640 READ X,Y,GR
650 CALL HCHAR(X,Y,GR)
660 NEXT I
```

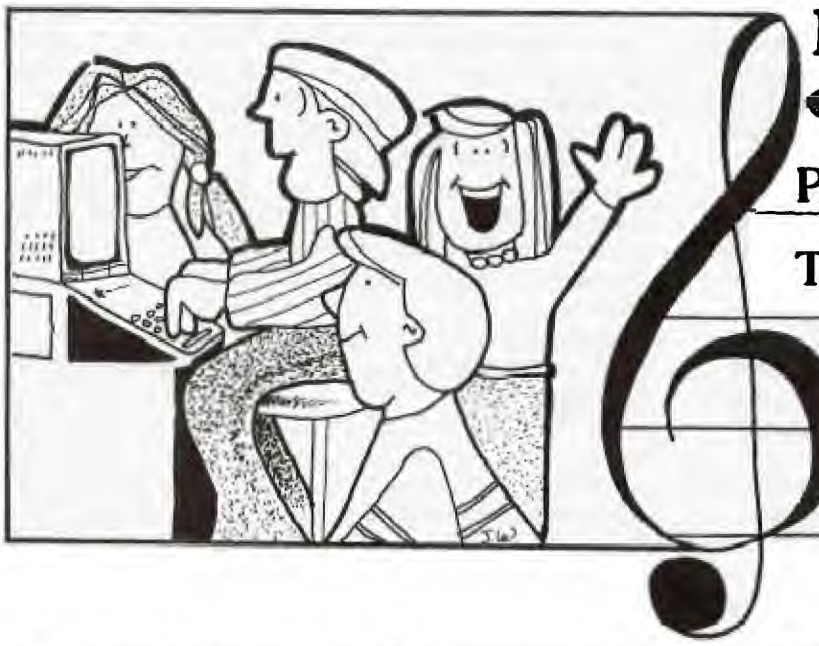
```
670 CALL SOUND(T,392,2)
680 DATA 3,8,146,3,9,147,4,9,140,5,9,1
    40,6,9,137,6,8,136,7,8,138,7,9,139
690 RESTORE 680
700 FOR I=1 TO 8
710 READ X,Y,GR
720 CALL HCHAR(X,Y,GR)
730 NEXT I
740 CALL SOUND(T,440,2)
750 CALL VCHAR(16,12,49)
760 DATA 75,69,89,66,79,65,82,68
770 RESTORE 760
780 FOR Y=14 TO 21
790 READ L
800 CALL HCHAR(16,Y,L)
810 NEXT Y
820 CALL SOUND(T,494,2)
830 DATA 4,26,112,5,26,113,6,26,114,6,
    27,115,7,26,116,7,27,117
840 RESTORE 830
850 FOR I=1 TO 6
860 READ X,Y,GR
870 CALL HCHAR(X,Y,GR)
880 NEXT I
890 CALL SOUND(T,523,2)
900 DATA 7,28,118,8,26,119,8,27,120,8,
    28,121,9,27,122
910 RESTORE 900
920 CALL SOUND(T,262,2)
930 FOR I=1 TO 5
940 READ X,Y,GR
950 CALL HCHAR(X,Y,GR)
960 NEXT I
970 CALL SOUND(T,330,2)
980 CALL HCHAR(18,12,50)
990 DATA 84,82,69,66,76,69,32,67,76,69
    ,70
1000 RESTORE 990
1010 FOR Y=14 TO 24
1020 READ L
1030 CALL HCHAR(18,Y,L)
1040 NEXT Y
1050 CALL SOUND(T,392,2)
1060 CALL SOUND(T,523,2)
1070 CALL HCHAR(20,12,51)
1080 DATA 66,65,83,83,32,67,76,69,70
1090 RESTORE 1080
1100 FOR Y=14 TO 22
1110 READ L
1120 CALL HCHAR(20,Y,L)
1130 NEXT Y
1140 CALL SOUND(T,392,2)
1150 DATA 18,27,138,18,28,139,17,27,136
    ,17,28,137,16,28,140,15,28,140,14,
    28,141,14,29,142,15,29,143
1160 RESTORE 1150
1170 FOR I=1 TO 9
1180 READ X,Y,GR
1190 CALL HCHAR(X,Y,GR)
1200 NEXT I
1210 CALL SOUND(T,330,2)
1220 CALL SOUND(T,262,2)
1230 GOTO 5430
1240 CALL CLEAR
1250 CALL COLOR(4,2,7)
1260 CALL COLOR(9,2,1)
1270 CALL COLOR(10,2,16)
1280 CALL CHAR(96,"FF")
1290 CALL CHAR(104,"FF")
1300 CALL CHAR(105,"7F7F7F7F7F7F7F7F")
1310 CALL CHAR(106,"FEFEFEFEFEFEFEFE")
1320 CALL CHAR(107,"0101010101010101")
1330 CALL CHAR(108,"8080808080808080")
1340 CALL CHAR(109,"0")
1350 RANDOMIZE
1360 COUNT=0
1370 KB=INT(2*RND)+1
1380 CALL CLEAR
1390 DATA 78,65,77,69,32,84,72,69,32,75
    ,69,89
1400 RESTORE 1390
1410 FOR Y=11 TO 22
1420 READ L
1430 CALL HCHAR(4,Y,L)
1440 NEXT Y
1450 PAT=KB
1460 REM DRAW KEYBOARD
1470 CALL HCHAR(8,1,104,32)
1480 CALL HCHAR(18,1,96,32)
1490 CALL VCHAR(8,1,109,10)
1500 CALL VCHAR(8,32,109,10)
1510 FOR Y=3 TO 30 STEP 3
1520 CALL VCHAR(8,Y-1,109,10)
1530 CALL VCHAR(8,Y,107,10)
1540 CALL VCHAR(8,Y+1,108,10)
1550 NEXT Y
1560 IF KB=2 THEN 1660
1570 REM KEYBOARD 1
1580 DATA 3,6,12,15,18,24,27,27
1590 RESTORE 1580
1600 FOR I=1 TO 8
1610 READ Y
1620 CALL VCHAR(8,Y,105,7)
1630 CALL VCHAR(8,Y+1,106,7)
1640 NEXT I
1650 GOTO 1740
1660 REM KEYBOARD 2
1670 DATA 3,6,9,15,18,24,27,30,30
1680 RESTORE 1670
1690 FOR I=1 TO 9
1700 READ Y
1710 CALL VCHAR(8,Y,105,7)
1720 CALL VCHAR(8,Y+1,106,7)
1730 NEXT I
1740 REM PICK KEY
1750 NN=INT(11*RND)+1
1760 J=3*NN-1
1770 CALL HCHAR(16,J,63)
1780 FOR I=1 TO 10
1790 CALL COLOR(4,2,7)
1800 CALL COLOR(4,16,7)
1810 NEXT I
1820 CALL KEY(0,NOTE,STATUS)
1830 IF STATUS<>1 THEN 1820
1840 IF KB=2 THEN 1860
1850 ON NN GOTO 2050,2120,2190,2260,233
    0,2400,2470,2540,2580,2620,2660
1860 ON NN GOTO 1890,1930,1970,2010,205
    0,2120,2190,2260,2330,2400,2470
1870 CALL SOUND(500,-4,2)
1880 GOTO 1820
1890 IF NOTE<>70 THEN 1870
1900 CALL SOUND(T,175,2)
1910 CALL HCHAR(16,2,70)
1920 GOTO 2690
1930 IF NOTE<>71 THEN 1870
1940 CALL SOUND(T,196,2)
1950 CALL HCHAR(16,5,71)
1960 GOTO 2690
1970 IF NOTE<>65 THEN 1870
1980 CALL SOUND(T,220,2)
1990 CALL HCHAR(16,8,65)
2000 GOTO 2690
2010 IF NOTE<>66 THEN 1870
2020 CALL SOUND(T,247,2)
2030 CALL HCHAR(16,11,66)
2040 GOTO 2690
2050 IF NOTE<>67 THEN 1870
2060 CALL SOUND(T,262,2)
2070 IF KB=2 THEN 2100
2080 CALL HCHAR(16,2,67)
2090 GOTO 2690
2100 CALL HCHAR(16,14,67)
2110 GOTO 2690
2120 IF NOTE<>68 THEN 1870
2130 CALL SOUND(T,294,2)
```

```
2140 IF KB=2 THEN 2170
2150 CALL HCHAR(16,5,68)
2160 GOTO 2690
2170 CALL HCHAR(16,17,68)
2180 GOTO 2690
2190 IF NOTE<>69 THEN 1870
2200 CALL SOUND(T,330,2)
2210 IF KB=2 THEN 2240
2220 CALL HCHAR(16,8,69)
2230 GOTO 2690
2240 CALL HCHAR(16,20,69)
2250 GOTO 2690
2260 IF NOTE<>70 THEN 1870
2270 CALL SOUND(T,349,2)
2280 IF KB=2 THEN 2310
2290 CALL HCHAR(16,11,70)
2300 GOTO 2690
2310 CALL HCHAR(16,23,70)
2320 GOTO 2690
2330 IF NOTE<>71 THEN 1870
2340 CALL SOUND(T,392,2)
2350 IF KB=2 THEN 2380
2360 CALL HCHAR(16,14,71)
2370 GOTO 2690
2380 CALL HCHAR(16,26,71)
2390 GOTO 2690
2400 IF NOTE<>65 THEN 1870
2410 CALL SOUND(T,440,2)
2420 IF KB=2 THEN 2450
2430 CALL HCHAR(16,17,65)
2440 GOTO 2690
2450 CALL HCHAR(16,29,65)
2460 GOTO 2690
2470 IF NOTE<>66 THEN 1870
2480 CALL SOUND(T,494,2)
2490 IF KB=2 THEN 2520
2500 CALL HCHAR(16,20,66)
2510 GOTO 2690
2520 CALL HCHAR(16,32,66)
2530 GOTO 2690
2540 IF NOTE<>67 THEN 1870
2550 CALL SOUND(T,523,2)
2560 CALL HCHAR(16,23,67)
2570 GOTO 2690
2580 IF NOTE<>68 THEN 1870
2590 CALL SOUND(T,587,2)
2600 CALL HCHAR(16,26,68)
2610 GOTO 2690
2620 IF NOTE<>69 THEN 1870
2630 CALL SOUND(T,659,2)
2640 CALL HCHAR(16,29,69)
2650 GOTO 2690
2660 IF NOTE<>70 THEN 1870
2670 CALL SOUND(T,698,2)
2680 CALL HCHAR(16,32,70)
2690 FOR DELAY=1 TO 1000
2700 NEXT DELAY
2710 CALL HCHAR(16,J,109)
2720 COUNT=COUNT+1
2730 IF COUNT=10 THEN 5400
2740 KB=INT(2*RND)+1
2750 IF KB=PAT THEN 1740 ELSE 1380
2760 REM TREBLE CLEF NOTES
2770 CALL CLEAR
2780 FOR I=9 TO 15
2790 CALL COLOR(I,2,1)
2800 NEXT I
2810 CALL CHAR(96,"FF")
2820 CALL CHAR(104,"0000007 1F3F78F0E0")
2830 CALL CHAR(105,"000000C0F0F07 83C")
2840 CALL CHAR(106,"C0808080808080808080")
2850 CALL CHAR(107,"1C0C0C0C0C0C0C040404")
2860 CALL CHAR(108,"8080808080808080 40")
2870 CALL CHAR(109,"0404040404080808 18")
2880 CALL CHAR(110,"FF4040404040202020")
2890 CALL CHAR(111,"FF3040606040404 0C080")
2900 CALL CHAR(112,"2121233312161C1C18")
2910 CALL CHAR(113,"FF0000001030 70F1E")
2920 CALL CHAR(114,"FF78E8C8848404040")
2930 CALL CHAR(115,"0000000010103070F")
2940 CALL CHAR(116,"3C78F0E0C0808080")
2950 CALL CHAR(117,"0404020202020202 02")
2960 CALL CHAR(118,"FF0E1C1838307070")
2970 CALL CHAR(119,"FF01070F1F3D3971")
2980 CALL CHAR(120,"FFFFFFC0")
2990 CALL CHAR(121,"FFFEFF0F0F0301")
3000 CALL CHAR(122,"FF000080C0C0E060")
3010 CALL CHAR(123,"6060606E0E060606 060")
3020 CALL CHAR(125,"60E0C0C0C0C0C0E0")
3030 CALL CHAR(126,"8080808080808080 80")
3040 CALL CHAR(127,"3030101010101010 10")
3050 CALL CHAR(128,"FF7070383833C1C1E")
3060 CALL CHAR(129,"FF6060303 10")
3070 CALL CHAR(130,"FF4040404040404040 40")
3080 CALL CHAR(131,"FF1020202040C080")
3090 CALL CHAR(132,"0F070301")
3100 CALL CHAR(133,"0080C0F0F83C0F03")
3110 CALL CHAR(134,"00000000000000C0")
3120 CALL CHAR(135,"2020202020202212E")
3130 CALL CHAR(136,"0103060810608 0")
3140 CALL CHAR(137,"FF2020202020202020")
3150 CALL CHAR(138,"1010101010101010")
3160 CALL CHAR(139,"3C7EFEFEFC7C391F")
3170 CALL CHAR(140,"1010103020408 0")
3180 CALL CHAR(141,"FF8080808080808080")
3190 CALL CHAR(152,"030F1F3F7FFFFFFF")
3200 CALL CHAR(153,"FCFFFFFFFFFFFFFF")
3210 CALL CHAR(154,"FFFFFF7F3F1F0F03")
3220 CALL CHAR(155,"FFFFFFFFFFFFFFFC")
3230 CALL CHAR(156,"F0F0F0E0C080 ")
3240 CALL CHAR(157,"1010909D0F0F0F0F0")
3250 CALL CHAR(158,"000080C0E0F0F0F0")
3260 CALL CHAR(159,"0101010101010101")
3270 CALL COLOR(16,7,1)
3280 GOSUB 4980
3290 DATA 78,65,77,69,32,84,72,69,32,78
     ,79,84,69
3300 RESTORE 3290
3310 FOR Y=10 TO 22
3320 READ L
3330 CALL HCHAR(6,Y,L)
3340 NEXT Y
3350 IF GAME=3 THEN 4040
3360 REM TREBLE CLEF
3370 DATA 9,5,104,9,6,105,10,5,106,10,6
     ,107,11,5,108,11,6,109,12,5,110,12
     ,6,111,13,5,112
3380 RESTORE 3370
3390 DATA 14,4,113,14,5,114,15,3,115,15
     ,4,116,15,5,117,16,3,118,16,5,119,
     16,6,120,16,7,121,16,8,122
3400 DATA 17,3,123,17,5,125,17,6,126,17
     ,8,127,18,3,128,18,5,129,18,6,130,
     18,8,131,19,3,132
3410 DATA 19,4,133,19,5,134,19,6,135,19
     ,7,136,20,6,137,21,6,138,22,5,139,
     22,6,140,1,1,32
3420 FOR I=1 TO 37
3430 READ X,Y,GR
3440 CALL HCHAR(X,Y,GR)
3450 NEXT I
3460 COUNT=0
3470 RANDOMIZE
3480 NN=INT(11*RND)+1
3490 M=22-NN
3500 GOSUB 5070
3510 IF NN<>1 THEN 3540
3520 CALL HCHAR(M+1,14,96)
3530 CALL HCHAR(M+1,18,96)
3540 CALL KEY(0,NOTE,STATUS)
3550 IF STATUS<>1 THEN 3540
3560 ON NN GOTO 3590,3630,3670,3710,375
     0,3790,3830,3870,3910,3950,3990
3570 CALL SOUND(600,-8,2)
3580 GOTO 3540
3590 IF NOTE<>67 THEN 3570
```

```
3600 CALL SOUND(T,262,2)
3610 CALL HCHAR(M+1,16,67)
3620 GOTO 4020
3630 IF NOTE<>68 THEN 3570
3640 CALL SOUND(T,294,2)
3650 CALL HCHAR(M+1,16,68)
3660 GOTO 4020
3670 IF NOTE<>69 THEN 3570
3680 CALL SOUND(T,330,2)
3690 CALL HCHAR(M+1,16,69)
3700 GOTO 4020
3710 IF NOTE<>70 THEN 3570
3720 CALL SOUND(T,349,2)
3730 CALL HCHAR(M+1,16,70)
3740 GOTO 4020
3750 IF NOTE<>71 THEN 3570
3760 CALL SOUND(T,392,2)
3770 CALL HCHAR(M+1,16,71)
3780 GOTO 4020
3790 IF NOTE<>65 THEN 3570
3800 CALL SOUND(T,440,2)
3810 CALL HCHAR(M+1,16,65)
3820 GOTO 4020
3830 IF NOTE<>66 THEN 3570
3840 CALL SOUND(T,494,2)
3850 CALL HCHAR(M+1,16,66)
3860 GOTO 4020
3870 IF NOTE<>67 THEN 3570
3880 CALL SOUND(T,523,2)
3890 CALL HCHAR(M+1,16,67)
3900 GOTO 4020
3910 IF NOTE<>68 THEN 3570
3920 CALL SOUND(T,587,2)
3930 CALL HCHAR(M+1,16,68)
3940 GOTO 4020
3950 IF NOTE<>69 THEN 3570
3960 CALL SOUND(T,659,2)
3970 CALL HCHAR(M+1,16,69)
3980 GOTO 4020
3990 IF NOTE<>70 THEN 3570
4000 CALL SOUND(T,698,2)
4010 CALL HCHAR(M+1,16,70)
4020 GOSUB 5220
4030 GOTO 3480
4040 CALL CHAR(97,"FF000000003078FC")
4050 CALL CHAR(98,"FC7830")
4060 CALL CHAR(99,"FFFF7E3C")
4070 CALL CHAR(100,"1810206060 7C7EFF")
4080 CALL CHAR(101,"FF00000103060408")
4090 CALL CHAR(102,"FF0F70C0")
4100 CALL CHAR(103,"FFC0381C0E070303")
4110 CALL CHAR(124,"C0C0603038383838")
4120 CALL CHAR(142,"FF1C1C1C1C1C1C1C")
4130 CALL CHAR(143,"1C1C1C1C1C1C1C1C")
4140 CALL CHAR(144,"FF38383870 70C0C0")
4150 CALL CHAR(145,"0303060 40C1870C0")
4160 CALL CHAR(146,"FF03060C106080")
4170 DATA 14,3,99,13,3,100,12,3,101,12,
     4,102,12,6,103,13,7,124,14,7,142
4180 RESTORE 4170
4190 DATA 15,7,143,16,7,144,17,6,145,18
     ,5,146,12,9,97,13,9,98,14,9,97,15,
     9,98,1,1,32
4200 FOR I=1 TO 16
4210 READ X,Y,GR
4220 CALL HCHAR(X,Y,GR)
4230 NEXT I
4240 COUNT=0
4250 RANDOMIZE
4260 NN=INT(11*RND)+1
4270 M=20-NN
4280 GOSUB 5070
4290 IF NN<>11 THEN 4320
4300 CALL HCHAR(M+1,14,96)
4310 CALL HCHAR(M+1,18,96)
4320 CALL KEY(0,NOTE,STATUS)
4330 IF STATUS<>1 THEN 4320
4340 ON NN GOTO 4370,4420,4470,4520,457
     0,4620,4670,4720,4770,4820,4870
4350 CALL SOUND(600,-8,2)
4360 GOTO 4320
4370 IF NOTE<>71 THEN 4350
4380 J=110
4390 GOSUB 4930
4400 CALL HCHAR(20,16,71)
4410 GOTO 4910
4420 IF NOTE<>65 THEN 4350
4430 J=116
4440 GOSUB 4930
4450 CALL HCHAR(19,16,65)
4460 GOTO 4910
4470 IF NOTE<>66 THEN 4350
4480 J=123
4490 GOSUB 4930
4500 CALL HCHAR(18,16,66)
4510 GOTO 4910
4520 IF NOTE<>67 THEN 4350
4530 J=131
4540 GOSUB 4930
4550 CALL HCHAR(17,16,67)
4560 GOTO 4910
4570 IF NOTE<>68 THEN 4350
4580 J=147
4590 GOSUB 4930
4600 CALL HCHAR(16,16,68)
4610 GOTO 4910
4620 IF NOTE<>69 THEN 4350
4630 J=165
4640 GOSUB 4930
4650 CALL HCHAR(15,16,69)
4660 GOTO 4910
4670 IF NOTE<>70 THEN 4350
4680 J=175
4690 GOSUB 4930
4700 CALL HCHAR(14,16,70)
4710 GOTO 4910
4720 IF NOTE<>71 THEN 4350
4730 J=196
4740 GOSUB 4930
4750 CALL HCHAR(13,16,71)
4760 GOTO 4910
4770 IF NOTE<>65 THEN 4350
4780 J=220
4790 GOSUB 4930
4800 CALL HCHAR(12,16,65)
4810 GOTO 4910
4820 IF NOTE<>66 THEN 4350
4830 J=247
4840 GOSUB 4930
4850 CALL HCHAR(11,16,66)
4860 GOTO 4910
4870 IF NOTE<>67 THEN 4350
4880 J=262
4890 GOSUB 4930
4900 CALL HCHAR(10,16,67)
4910 GOSUB 5220
4920 GOTO 4260
4930 FOR I=1 TO 5
4940 CALL SOUND(200,J,2)
4950 J=J+15
4960 NEXT I
4970 RETURN
4980 REM STAFF
4990 FOR X=12 TO 20 STEP 2
5000 CALL HCHAR(X,1,96,31)
5010 NEXT X
5020 FOR X=12 TO 18 STEP 2
5030 CALL HCHAR(X,1,141)
5040 CALL HCHAR(X+1,1,126)
5050 NEXT X
5060 RETURN
5070 REM NOTE
5080 CALL HCHAR(M,15,152)
5090 CALL HCHAR(M,16,153)
5100 CALL HCHAR(M+1,15,154)
5110 CALL HCHAR(M+1,16,155)
5120 CALL HCHAR(M+1,17,156)
```

```
5130 IF GAME=3 THEN 5210
5140 IF NN<8 THEN 5180
5150 CALL HCHAR(M,17,158)
5160 CALL VCHAR(M+1,14,159,6)
5170 RETURN
5180 CALL HCHAR(M,17,157)
5190 CALL VCHAR(M-5,17,138,5)
5200 RETURN
5210 IF NN<5 THEN 5180 ELSE 5150
5220 REM CHANGE NOTES
5230 FOR DELAY=1 TO 1000
5240 NEXT DELAY
5250 COUNT=COUNT+1
5260 IF COUNT=10 THEN 5400
5270 CALL HCHAR(M,14,32,5)
5280 CALL HCHAR(M+1,14,32,5)
5290 IF GAME=3 THEN 5350
5300 IF NN<8 THEN 5330
5310 CALL VCHAR(M+1,14,32,6)
```

```
5320 GOTO 5360
5330 CALL VCHAR(M-5,17,32,5)
5340 GOTO 5360
5350 IF NN<5 THEN 5330 ELSE 5310
5360 FOR K=12 TO 20 STEP 2
5370 CALL HCHAR(K,14,96,5)
5380 NEXT K
5390 RETURN
5400 PRINT "PRESS 1 FOR KEYBOARD QUIZ"
5410 PRINT TAB(7);"2 FOR TREBLE NOTES"
5420 PRINT TAB(7);"3 FOR BASS NOTES"
5430 PRINT TAB(7);"4 TO END PROGRAM"
5440 CALL KEY(0,K,S)
5450 IF K=52 THEN 5500
5460 IF K>51 THEN 5440
5470 IF K<49 THEN 5440
5480 GAME=K-48
5490 IF GAME=1 THEN 1240 ELSE 2770
5500 END
```



Notes on a Computer Score: Part 1 - The TI-99/4 Conducts Music Theory Drill in a Traditional Classroom Setting.

Recently I returned to my job as elementary music teacher is the Rossford (Ohio) School District after an exciting and rewarding summer. When reading students' responses to the question, "What did you most enjoy about music last year?" on a first-day questionnaire, I was pleased to see the number of students responding, "The computer." At New Horizons Academy for the Gifted, a Computer-Assisted Music Program held in the summer of 1981 elicited a similar response from students. In both of these very different educational contexts, computer usage has proved to be a strong motivational force in students' acquisition of music theory and skills.

Last year my husband and I purchased a TI-99/4 with some rather nebulous ideas about potential applications in the general music curriculum. Together we worked on the development of programs—trying to incorporate motivational strategies which apply in virtually any teaching situation—and experimented with various uses of commercially available software (e.g., the *Music Maker* Command Cartridge and *Music Skills Trainer*). We tried many techniques in the classroom to determine the children's responses. The result has been the continuing evolution of a computer-assisted music curriculum tailored to the needs of my teaching situation. It has been a stimulating year—one which progressed from ignorance about using the computer and apprehensions concerning its effectiveness to one of the most exciting experiences of my teaching career.

Whether you are a teacher planning a Computer-Assisted Instruction (CAI) project or a parent considering the potential educational value of a computer in your home, it may not be necessary to know exactly where you are going before you take the first step. Our experience has shown that the element of discovery inherent in developing a curriculum interactively with children can be as rewarding and exciting a learning process for the educator as is the use of the final product for the student.

Glenwood School in the Rossford (Ohio) district is a typical elementary school with an enrollment of 400 students in grades 1-6. I incorporated the computer into my general music curriculum for grades 4-6 during the fall semester of the 1980-81 school year (my first year in this system). Classes were intact groups which met for two 35-minute periods each week in the "music room"—a corner of the cafeteria. Average class size was 25 students, with pupils from the Adjusted Curriculum (learning difficulties) as well as Project Horizons (gifted) programs mainstreamed into the regular classes

My classes are organized around the belief that music should be fun and provide students with an outlet for their creativity. Although music class can be a break from routine academics, children must be equipped with basic knowledge of the fundamentals of music reading and theory upon completion of a general music course. A variety of experiences —singing, movement, listening and playing instruments— should be provided. The computer was employed as an additional enrichment activity—one which turned out to be unusually effective for the students, as well as challenging for the teacher.

The general approach I employ involves an initial experiential emphasis (e.g., singing and movement) followed by instruction in the basic theory required to read music. In addition to providing knowledge of theory, this approach readies students for potential participation in band and choir.

Two computer programs, *Rhythm* and *Mystery Words,* were used to reinforce two aspects of the curriculum: (1) aural recognition of rhythmic patterns, and (2) knowledge of musical notation for note names in both treble and bass clef.

In teaching students how to discriminate between various note values and rests, I first used "Echo Clapping" in which I clapped a rhythmic pattern and the students tried to reproduce it. Next, students were taught to associate appropriate music terms with relative durations (whole, half, quarter, eighth and their corresponding rests).

Then we progressed to an activity called "Rhythmic Dictation" in which students wrote in musical notation the rhythms they heard clapped (e.g., ♩♩♩♩,♩𝅘𝅥𝅯𝅘𝅥𝅯𝅘𝅥𝅯𝅘𝅥𝅯,♩♩♫♫♩). So as not to get too complex at the beginning, only the ♩,♫ and 𝅘𝅥𝅯𝅘𝅥𝅯 were taught.

Following presentation of the concepts and introduction to initial rhythmic dictation exercises, many students reach

a plateau when acquiring the skill of describing rhythmic patterns in musical notation. Representation of rhythmic patterns by clapping is to some extent an abstraction because it requires analysis of the relationship between a steady beat and intervals between claps. Although most children are highly motivated to acquire this skill, many encounter difficulties which result, in part, from its abstract nature.

To address this problem, we tried to incorporate two principles into the *Rhythm* program: (1) concreteness, and (2) immediacy of feedback. At the same time, in order to optimize the effect of one computer on 25 children, we decided to use a game format—allowing for the possibility of up to ten teams within a class. We also put a lot of effort into the introductory portion of the program to catch students' attention with the color and sound capabilities of the TI-99/4.

I initally introduced this program in my sixth grade classes after considerable time had been spent practicing rhythmic dictation. Periodic quizzes showed that a fair number of students in all three classes had not grasped the concept. After three sessions with the *Rhythm* program, almost every student had become competent in clapped rhythmic dictation.

I believe several factors contributed to this remarkable improvement: The activity was conceptually more concrete than rhythmic dictation, and it provided students with immediate feedback which included the correct response when mistakes were made. Concentration and motivation were improved, in part, simply by the uniqueness of the computer activity. It was not uncommon for several students to show up in the music room shortly after their bus arrived, in the hope of spending ten minutes working with a computer music game before school started. When teachers arrived after a general music class to take their students back to their classrooms, we invariably had to pry some of them away; classes sometimes had to be actually extended 10 to 15 minutes!

The group dynamics involved also played a significant role. During the first session, the sixth graders asked to have team scores added together for comparison with the other sixth grade classes. Because each class represented an intact group with some history and cohesiveness, a positive atmosphere prevailed in which the student working at the computer was supported and encouraged by the cheers and comments of fellow classmates. All students had several opportunities to make a contribution to the class total score.

Next, I used the *Rhythm* program in the fifth grade. Most students were aware through word-of-mouth that the sixth graders had been using a computer and naturally were interested in the top score the sixth graders had achieved. The typical score for the first day in the sixth grade had been 15 to 20. By comparison, the first day scores in the fifth grade were as high as 45! Similar enthusiam was observed in the fourth grade.

For the most part, I anticipated these outcomes, although the actual impressive results far exceeded my expectations. There were also some genuine surprises: First, using the computer allowed me to observe and diagnose the problems of individual students and, where necessary, to take them aside and give special attention to their needs while the class was occupied with the computer. Learning the concept was important to them in order to make points for their class.

Another gratifying result was that the students with learning difficulties found it easier to master this more concrete activity and took a great deal of pride in their contributions to the class score. It was indeed rewarding to see their beaming faces as classmates cheered and patted them on the back after their correct responses.

Following the computer activity, nearly every student had achieved competency in the basic rhythms which had been presented, and they were able to apply this knowledge in the playing of rhythm instruments. I wrote several lines of rhythmic patterns on the board in musical notation and asked individuals or small groups to play them. Subsequently, the patterns were played to accompany class singing or listening to records.

The rhythm unit was followed by the study of musical notation for pitch. Students were introduced to this concept through a discussion of the importance of learning the note names on the staff in order to read music when singing or playing instruments. I compared note reading to reading a foreign language, using symbols and notes instead of words to create a musical story.

Initial instruction presented the familiar phrases "Every Good Boy Does Fine" and "F A C E" to facilitate learning the position of notes in the treble clef, and this was followed by drills to further reinforce note name recognition. Students were promised that the computer would be brought back to class when they learned these note names well enough to play a computer game. Thereafter, the *Mystery Words* game listed at the end of this article was introduced.

Mystery Words is a game that is based upon the use of note name letters to spell a variety of words, for example,"cabbage," "bead," and "facade." The program randomly chooses one of these words and represents it in music notation graphics in the treble clef, the bass clef, or both. The teacher has the option of excluding words with more difficult meanings (such as "facade" or "accede") when using the game with younger students.

The screen is divided in half with a red side and a blue side corresponding to the red and blue teams into which the class is divided. One member of each team is seated at the console. Before the presentation of a Mystery Word each player must signal he is ready by pressing the "1" or "0" key. As each team member signals readiness, a traffic light on each side of the screen changes from red to yellow to green, the Mystery Theme is played, and the graphic representation of the Mystery Word appears. The first student to decipher the word presses the 1 or 0 key, and the graphic representation disappears. He is then instructed to enter the answer using the keys 3 through 9 which have been labeled A through G on a blank keyboard overlay. As each letter is pressed, it appears in the graphics window. If the entire word is entered correctly, the graphic representation reappears with notes above the letters entered by the student, and the team's score is incremented. In the event an incorrect letter is entered, the opposing team member is instructed to try.

When the game was introduced in class, only the treble clef option was selected since previous instruction did not include the bass clef. Prior experience suggested that the presentation of both treble and bass clefs was too confusing for the average elementary age student. But using the

computer, students quickly mastered treble clef note names and requested that they be allowed to try working with the bass clef as well. Their ability to learn bass clef note names rapidly with minimal prior classroom work and to work with both clefs simultaneously was truly amazing.

Use of *Mystery Words* was accompanied by the same intense interest and motivation as *Rhythm*. Although this game was also constructed for intra-class competition, students again asked that team scores be added together for comparison with other classes. Some students began showing up before school with younger brothers and sisters to explain the computer games to them, and I became a popular figure among students in the cafeteria at lunch time—the main topic of conversation being the computer.

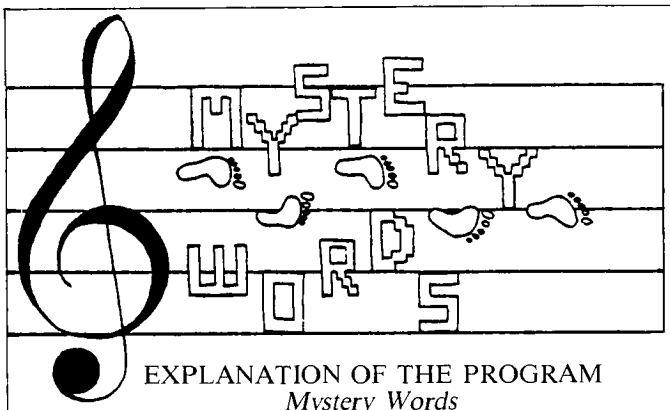
Seymour Papert defines three components for learning mathematics: the *Continuity Principle*, the *Power Principle* and the *Principle of Cultural Resonance*.* These principles, of course, may be applied to the acquisition of any content domain—not just mathematics. This is to say that a concept or skill may be acquired with the least effort if it (1) is continuous with what the learner already knows, (2) empowers him to achieve personal objectives which could not be achieved otherwise, and (3) makes sense within a larger social context. Construing the computer-assisted units on these principles may help to elucidate some of the elements which I believe are critical to the success of this application (and for that matter, of any learning environment).

All children are intimately familar with music in their everyday environment. The initial emphasis on those aspects of music already familar to them, i.e., singing and movement, helped to give a sense of continuity with respect to subsequent course material. Second, the computer activity was integrated into a larger social context by the students themselves when they asked that team scores be added together for comparison with other classes. This phenomenon was also apparent when students in lower grades expressed interest in the scores obtained by the upper grades and used this information in the setting of personal goals.

But perhaps the most important element is the *Power Principle*. Students perceived that the acquisiton of music skills would enable them to make contributions toward group achievement in the computer music game. Later they found that they were able to play rhythm instruments and read musical notation, and at the same time, they realized that these skills may be useful to them in the future when participating in band and chorus activities, which are themselves part of a meaningful social context.

In summary, use of the computer increased student motivation, and at the same time allowed abstract material to be represented in a concrete way, leading to more rapid acquisition of skills and concepts. The computer also made it possible to diagnose individual weaknesses and provide individualized remedial work. Discipline problems arising from boredom and lack of interest were nonexistent when the computer was in the classroom. In general, a positive environment, cooperation, and mutual support predominated.

*Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. New York: Basic Books, 1980.



EXPLANATION OF THE PROGRAM
Mystery Words

Line Nos.

130-200	Clears screen; sets screen color.
210-270	Accepts user input for word list.
280-350	Accepts user choice of type of drill.
360-420	Instruction for keyboard labeling.
430-490	Interrupts program for explanation to class.
500-700	Displays notes of treble clef for review.
710-840	Displays notes of bass clef for review.
850-870	Restores data pointer and dimensions arrays.
888-980	Defines character patterns and sets colors for initial graphic screen.
990-1100	Displays initial graphics screen; plays associated sounds.
1110-1330	Displays *Mystery Words* title screen.
1340-1390	Assigns word codes to SET$ array; numeric digits correspond to letters (1 = A, 2 = B, etc.).
1400-1470	Assigns coded print locations (potentially 2 for each staff) for each of the 7 letter codes to array LINE.
1480-1920	Defines character patterns and sets colors used in the game screen.
1930-2090	Displays basic elements of game screen.
2100-2110	Selects a Mystery Word randomly.
2120-2530	Determines a representation for word on staff(s); when more than 1 possibility for a location exists, choice is random. The word is played in invisible characters (white on white).
2540-3000	Displays READY message, changes traffic light colors as player signal ready by pressing keys 1 and 0; plays Mystery Theme.
3010-3030	Makes representation of Mystery Word visible.
3040-3100	Accepts answer signals from players.
3110-3180	Flashes screen of red team.
3190-3250	Flashes screen of blue team.
3260-3280	Removes Mystery Word from view pending player's response.
3290-3300	Instructs team to answer.
3310-3450	Accepts key 3-9 (A-G) input, checking each letter against correct spelling.
3460-3690	If answer is correct, displays Mystery Word again with spelling; increments team score; returns to line 1930.
3700-3820	If answer is incorrect, allows opposing team to respond.

Subroutines:

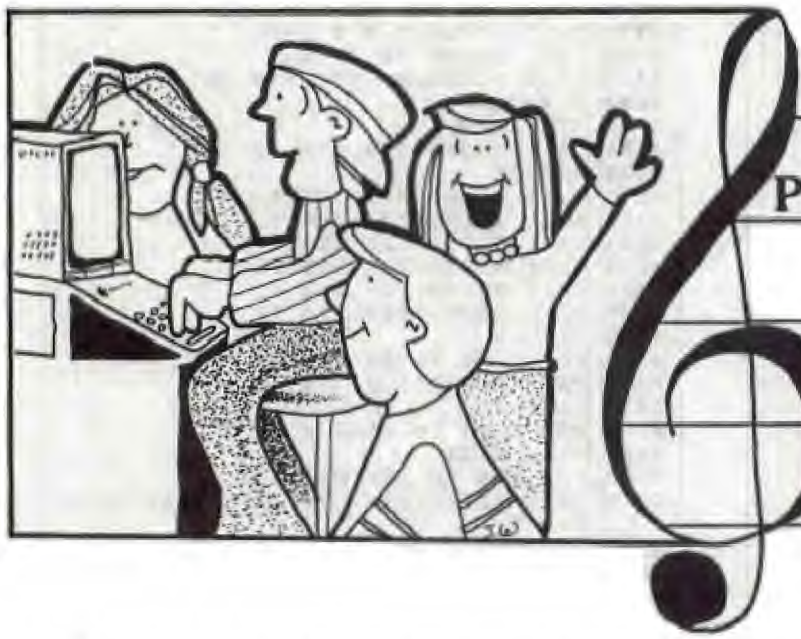
3830-4330	Displays game screen.
4340-4470	Draws treble staff.
4480-4570	Draws bass staff.
4580-4630	Displays messages, MSG$.
4640-4700	Displays letters of answer.
4710-4830	Displays initial graphic screen.
4840-4900	Plays Mystery Theme.
4910-5140	Displays treble and/or bass clef notes for initial review.
5160-5180	Erases screens after initial user input.

```
100 REM      MYSTERY WORDS
110 REM
120 REM
130 CALL CLEAR
140 NSET=52
150 RANDOMIZE
160 CALL SCREEN(5)
170 CALL VCHAR(1,31,1,96)
180 FOR I=1 TO 8
190 CALL COLOR(I,2,8)
200 NEXT I
210 PRINT " SOME OF THE WORDS IN THE
    WORD LIST ARE NOT READILY UNDE
    RSTOOD BY YOUNGER CHIL- DREN.":::
220 PRINT " WOULD YOU LIKE TO EXCLUDE
    THESE WORDS (Y/N)?":::::::
    :
230 CALL KEY(0,K,S)
240 IF K=89 THEN 270
250 IF K<>78 THEN 230
260 NSET=59
270 CALL HCHAR(16,23,K)
280 GOSUB 5150
290 PRINT " THREE DRILLS ARE AVAILABLE
    ":::::TAB(5);"1) TREBLE CLEF":
    TAB(5);"2) BASS CLEF"::
300 PRINT TAB(5);"3) TREBLE AND BASS":
    :::::" YOUR CHOICE (1,2,OR 3)?"
    :::::
310 CALL KEY(0,K,S)
320 IF K<49 THEN 310
330 IF K>51 THEN 310
340 CLEF=K-48
350 CALL HCHAR(21,28,K)
360 GOSUB 5150
370 PRINT TAB(9);"INSTRUCTIONS":"LAB
    EL 3-9 AS A-G, 1 RED AND 0 BLUE WI
    TH BLANK OVERLAY."
380 PRINT :"BOTH 1 AND 0 MUST BE PRESS
    EDAFTER TRAFFIC LIGHT IS RED."
390 PRINT :"FIRST PLAYER TO DECODE WOR
    D PRESSES 1/0 AND USES 3-9 TO ENTE
    R THE ANSWER."
400 PRINT ::::::TAB(8);"PRESS ANY K
    EY"
410 CALL KEY(0,K,S)
420 IF S=0 THEN 410
430 GOSUB 5150
440 MSG$="PRESS ANY KEY TO BEGIN"
450 ROW=12
460 COL=6
470 GOSUB 4600
480 CALL KEY(0,K,S)
490 IF S=0 THEN 480
500 CALL CLEAR
510 FOR I=1 TO 8
520 CALL COLOR(I,2,16)
530 NEXT I
540 RESTORE 5110
550 GOTO 1480
560 CALL CHAR(119,"000000FFFF")
570 ON CLEF GOTO 580,710,580
580 GOSUB 4340
590 ROW=4
600 COL=12
610 MSG$="TREBLE CLEF"
620 GOSUB 4600
630 ROW=23
640 COL=10
650 F1=4
660 S1=24
670 F2=20
680 S2=39
690 GOSUB 4910
700 IF CLEF<>3 THEN 850
710 CALL CLEAR
720 R=2
730 GOSUB 4480
740 ROW=4
750 COL=13
760 MSG$="BASS CLEF"
770 GOSUB 4600
780 ROW=23
790 COL=10
800 F1=1
810 S1=21
820 F2=14
830 S2=33
835 RESTORE 5130
840 GOSUB 4910
850 RESTORE
860 OPTION BASE 1
870 DIM SET$(59),NLINE(3),LINE(7,2,2)
880 CALL CLEAR
890 FLAG=1
900 CALL CHAR(35,"FFFFFFFFFFFFFFFF")
910 CALL CHAR(40,"")
920 CALL CHAR(96,"3C7E7E7E7E7E7E7E")
930 CALL CHAR(97,"3C3C000003C3C3C18")
940 CALL CLEAR
950 CALL COLOR(1,6,1)
960 CALL SCREEN(2)
970 CALL COLOR(2,7,7)
980 CALL COLOR(9,11,2)
990 R=24
1000 M=2
1010 MM=500
1020 FOR I=1 TO 1000
1030 NEXT I
1040 GOSUB 4710
1050 GOSUB 4840
1060 M=4
1070 MM=275
1080 GOSUB 4710
1090 GOSUB 4840
1100 CALL SOUND(1200,262,1,311,1,392,1)
1110 CALL CLEAR
1120 FOR I=1 TO 500
1130 NEXT I
1140 FOR I=1 TO 99
1150 READ D$
1160 CALL HCHAR(VAL(SEG$(D$,1,2)),VAL(S
     EG$(D$,3,2)),40)
1170 NEXT I
1180 FOR I=100 TO 188
1190 READ D$
1200 CALL HCHAR(VAL(SEG$(D$,1,2)),VAL(S
     EG$(D$,3,2)),35)
1210 NEXT I
1220 DATA 0503,0603,0703,0803,0903,1003
     ,1103,0604,0705,0606,0507,0607,070
     7,0807,0907,1007
1230 DATA 1107,0107,0207,0308,0111,0211
     ,0310,0409,0509,0609,0709,0809,071
     4,0713,0712,0711
1240 DATA 0811,0911,1011,1012,1013,1014
     ,1114,1214,1314,1313,1312,1311,051
     4,0515,0516,0517
1250 DATA 0518,0616,0716,0816,0916,1016
     ,1116,0721,0720,0719,0718,0818,091
     8,1018,1118,1218
1260 DATA 1318,1319,1320,1321,1019,1020
     ,0523,0623,0723,0823,0923,1023,112
     3,0524,0525,0526
1270 DATA 0626,0726,0826,0825,0824,0924
     ,1025,1126,0126,0226,0327,0130,023
     0,0329,0428,0528,0628
1280 DATA 0728,0828,1504,1604,1704,1804
     ,1904,2004,2104,2005,1906,2007,210
     8,2008,1908,1808
1290 DATA 1708,1608,1508,1813,1812,1811
     ,1810,1910,2010,2110,2210,2310,241
     0,2411,2412,2413
1300 DATA 2313,2213,2113,2013,1913,1615
     ,1715,1815,1915,2015,2115,2215,161
     6,1617,1618,1718
```

```
1310 DATA 1818,1918,1917,1916,2016,2117
     ,2218,1921,2021,2121,2221,2321,182
     0,1821,1822,1823
1320 DATA 1824,1924,2024,2124,2224,2324
     ,2424,2423,2422,2421,2420,1729,172
     8,1727,1726,1826
1330 DATA 1926,2026,2027,2028,2029,2129
     ,2229,2329,2328,2327,2326
1340 FOR I=1 TO 59
1350 READ SET$(I)
1360 NEXT I
1370 DATA 175,135,144,125,214,211,217,2
     55,257,254,251,312,412,414,452,455
     ,522,577,612,614,654,717,755,712,1
     754
1380 DATA 2125,2514,2556,3165,3175,4514
     ,4516,4554,5475,6135,6145,6554,717
     5,655,31754,45613 5,45314 5,45314
     ,1335 45,14175,21741 4,4531,613145,5
     66135
1400 FOR I=1 TO 2
1410 FOR J=1 TO 7
1420 FOR K=1 TO 2
1430 READ LINE(J,I,K)
1440 NEXT K
1450 NEXT J
1460 NEXT I
1470 DATA 5,0,-5,0,4,0,-4,0,3,-7,-3,6,-
     6,0,-3,6,-6,0,5,0,-5,0,4,0,-4,0,3,
     -7
1480 CALL CHAR(96,"0406050404040404")
1490 CALL CHAR(40,"")
1500 CALL CHAR(97,"0000000804020101")
1510 CALL CHAR(98,"0101010101010101")
1520 CALL CHAR(99,"FF0404040404040404")
1530 CALL CHAR(100,"FF0808081010204080")
1540 CALL CHAR(101,"FF0000000010204080")
1550 CALL CHAR(102,"FF061C6484040404")
1560 CALL CHAR(103,"FF")
1570 CALL CHAR(104,"FF1010101010101")
1580 CALL CHAR(105,"FF0F3444848484444")
1590 CALL CHAR(106,"FF00C0201010080")
1600 CALL CHAR(107,"FF0808040402010")
1610 CALL CHAR(108,"FF0404040404847F")
1620 CALL CHAR(109,"FF0808080102040808")
1630 CALL CHAR(110,"FF0000030408081")
1640 CALL CHAR(111,"FF00FC03")
1650 CALL CHAR(112,"FF0000038340202")
1660 CALL CHAR(113,"FF100804")
1670 CALL CHAR(114,"FF1010101313101")
1680 CALL CHAR(115,"FF0000000000001002")
1690 CALL CHAR(116,"FF2020404080")
1700 CALL CHAR(117,"FF0000000000106080")
1710 CALL CHAR(118,"FF04182 0C")
1720 CALL CHAR(120,"FF0000000038448 2")
1730 CALL CHAR(121,"FF824438")
1740 CALL CHAR(119,"FF3844828282443 8")
1750 CALL CHAR(122,"003844828282443 8")
1760 CALL CHAR(123,"0000000000038448 2")
1770 CALL CHAR(136,"")
1780 CALL CHAR(137,"3C7EFFFFFFFF7E3C")
1790 CALL CHAR(138,"FFFFFFFFFFFFFFFF")
1800 CALL CHAR(128,"3C7EFFFFFFFF7E3C")
1810 CALL CHAR(95,"")
1820 FOR I=5 TO 12
1830 CALL COLOR(I,2,16)
1840 NEXT I
1850 IF FLAG=0 THEN 560 ELSE 1860
1860 CALL COLOR(13,5,5)
1870 CALL COLOR(2,7,7)
1880 CALL COLOR(3,2,16)
1890 CALL COLOR(4,2,16)
1900 CALL COLOR(13,15,1)
1910 CALL COLOR(14,15,1)
1920 GOSUB 3830
1930 C=13
1940 TRY=1
1950 FOR I=5 TO 12
1960 CALL COLOR(I,16,16)
1970 NEXT I
1980 CALL HCHAR(2,13,95,13)
1990 FOR I=3 TO 7
2000 CALL HCHAR(I,13,103,13)
2010 NEXT I
2020 CALL HCHAR(9,13,95,13)
2030 IF CLEF<>3 THEN 2080
2040 CALL HCHAR(10,13,95,13)
2050 FOR I=11 TO 15
2060 CALL HCHAR(I,13,103,13)
2070 NEXT I
2080 CALL HCHAR(20,3,95,12)
2090 CALL HCHAR(20,19,95,12)
2100 N=INT(NSET*RND)+1
2110 WORD$=SET$(N)
2120 FOR I=1 TO LEN(WORD$)
2130 N=1
2140 FOR K=1 TO 3
2150 NLINE(K)=0
2160 NEXT K
2170 IF CLEF=2 THEN 2250
2180 FOR J=1 TO 2
2190 L=LINE(VAL(SEG$(WORD$,I,1)),1,J)
2200 IF L=0 THEN 2230
2210 NLINE(N)=L
2220 N=N+1
2230 NEXT J
2240 IF CLEF=1 THEN 2330
2250 FOR J=1 TO 2
2260 L=LINE(VAL(SEG$(WORD$,I,1)),2,J)
2270 IF L=0 THEN 2320
2280 IF CLEF=2 THEN 2300
2290 L=SGN(L)*(ABS(L)+8)
2300 NLINE(N)=L
2310 N=N+1
2320 NEXT J
2330 N=N-1
2340 IF N=1 THEN 2360
2350 N=INT(N*RND)+1
2360 L=NLINE(N)
2370 IF L<0 THEN 2450
2380 IF L=2 THEN 2420
2390 IF L=10 THEN 2420
2400 CALL HCHAR(L,C,119)
2410 GOTO 2430
2420 CALL HCHAR(L,C,122)
2430 C=C+2
2440 GOTO 2530
2450 L=ABS(L)
2460 IF L=3 THEN 2500
2470 IF L=11 THEN 2500
2480 CALL HCHAR(L-1,C,120)
2490 GOTO 2510
2500 CALL HCHAR(L-1,C,123)
2510 CALL HCHAR(L,C,121)
2520 C=C+2
2530 NEXT I
2540 ROW=20
2550 MSG$="READY"
2560 COL=3
2570 GOSUB 4600
2580 COL=19
2590 GOSUB 4600
2600 CALL COLOR(13,15,1)
2610 CALL HCHAR(12,3,137)
2620 CALL HCHAR(12,30,137)
2630 CALL HCHAR(8,3,128)
2640 CALL HCHAR(8,30,128)
2650 CALL COLOR(13,7,1)
2660 CALL SOUND(100,880,1)
2670 FOR I=5 TO 8
2680 CALL COLOR(I,2,16)
2690 NEXT I
2700 CALL KEY(1,KEY1,S)
2710 CALL KEY(2,KEY2,S)
```

```
2720 IF KEY1=19 THEN 2750
2730 IF KEY2=10 THEN 2850
2740 GOTO 2700
2750 CALL COLOR(13,15,1)
2760 CALL HCHAR(8,3,137)
2770 CALL HCHAR(8,30,137)
2780 CALL HCHAR(10,3,128)
2790 CALL HCHAR(10,30,128)
2800 CALL COLOR(13,11,1)
2810 CALL SOUND(100,880,1)
2820 CALL KEY(2,KEY2,S)
2830 IF KEY2<>10 THEN 2820
2840 GOTO 2940
2850 CALL COLOR(13,15,1)
2860 CALL HCHAR(8,3,137)
2870 CALL HCHAR(8,30,137)
2880 CALL HCHAR(10,3,128)
2890 CALL HCHAR(10,30,128)
2900 CALL COLOR(13,11,1)
2910 CALL SOUND(100,880,1)
2920 CALL KEY(1,KEY1,S)
2930 IF KEY1<>19 THEN 2920
2940 CALL COLOR(13,15,1)
2950 CALL HCHAR(10,3,137)
2960 CALL HCHAR(10,30,137)
2970 CALL HCHAR(12,3,128)
2980 CALL HCHAR(12,30,128)
2990 CALL COLOR(13,13,1)
3000 GOSUB 4840
3010 FOR I=9 TO 12
3020 CALL COLOR(I,2,16)
3030 NEXT I
3040 CALL KEY(1,KEY1,S1)
3050 CALL KEY(2,KEY2,S2)
3060 IF S1=-1 THEN 3040
3070 IF S2=-1 THEN 3040
3080 IF KEY1=19 THEN 3110
3090 IF KEY2=10 THEN 3190
3100 GOTO 3040
3110 CALL SOUND(300,-3,1)
3120 FOR M=1 TO 3
3130 CALL COLOR(2,11,11)
3140 CALL COLOR(2,7,7)
3150 NEXT M
3160 COL=3
3170 TEAM$="RED"
3180 GOTO 3260
3190 CALL SOUND(300,-2,1)
3200 FOR M=1 TO 3
3210 CALL COLOR(1,11,11)
3220 CALL COLOR(1,5,1)
3230 NEXT M
3240 COL=19
3250 TEAM$="BLUE"
3260 FOR I=9 TO 12
3270 CALL COLOR(I,16,16)
3280 NEXT I
3290 MSG$="OK_"&TEAM$&"_TEAM"
3300 GOSUB 4580
3310 C=13
3320 N=1
3330 CALL KEY(0,KEY,S)
3340 IF S<>1 THEN 3330
3350 IF KEY<51 THEN 3330
3360 IF KEY>57 THEN 3330
3370 KEY=KEY-50
3380 IF KEY<>VAL(SEG$(WORD$,N,1))THEN 3
700
3390 CALL SOUND(100,880,1)
3400 KEY=KEY+64
3410 CALL HCHAR(9,C,KEY)
3420 IF N=LEN(WORD$)THEN 3460
3430 N=N+1
3440 C=C+2
3450 GOTO 3330
3460 FOR I=9 TO 12
3470 CALL COLOR(I,2,16)
3480 NEXT I
3490 CALL SOUND(100,330,1,392,1,524,1)
3500 CALL SOUND(500,330,1,392,1,524,1)
3510 MSG$="RIGHT"
3520 IF TEAM$="BLUE" THEN 3550
3530 COL=3
3540 GOTO 3560
3550 COL=19
3560 GOSUB 4580
3570 IF TEAM$<>"RED" THEN 3620
3580 RED=RED+1
3590 C=3
3600 SCORE=RED
3610 GOTO 3650
3620 BLUE=BLUE+1
3630 C=29
3640 SCORE=BLUE
3650 MSG$=STR$(SCORE)
3660 GOSUB 4640
3670 FOR I=1 TO 500
3680 NEXT I
3690 GO TO 1930
3700 CALL SOUND(100,110,1)
3710 IF TRY=2 THEN 1930
3720 TRY=TRY+1
3730 CALL HCHAR(9,13,95,13)
3740 IF TEAM$="RED" THEN 3780
3750 TEAM$="RED"
3760 COL=3
3770 GOTO 3800
3780 TEAM$="BLUE"
3790 COL=19
3800 MSG$="YOU_TRY_"&TEAM$
3810 GOSUB 4580
3820 GOTO 3310
3830 CALL SCREEN(2)
3840 FOR I=1 TO 16
3850 CALL VCHAR(1,I,40,24)
3860 NEXT I
3870 FOR I=17 TO 32
3880 CALL VCHAR(1,I,35,24)
3890 NEXT I
3900 FOR I=2 TO 4
3910 CALL HCHAR(I,2,95,4)
3920 CALL HCHAR(I,28,95,4)
3930 NEXT I
3940 FOR I=19 TO 21
3950 CALL HCHAR(I,2,95,14)
3960 CALL HCHAR(I,18,95,14)
3970 NEXT I
3980 CALL HCHAR(1,7,95,20)
3990 CALL HCHAR(2,7,95,20)
4000 FOR I=3 TO 7
4010 CALL HCHAR(I,8,103,19)
4020 NEXT I
4030 FOR I=8 TO 10
4040 CALL HCHAR(I,7,95,20)
4050 NEXT I
4060 CALL HCHAR(7,7,95)
4070 CALL VCHAR(3,7,98,4)
4080 IF CLEF=2 THEN 4110
4090 GOSUB 4340
4100 GOTO 4140
4110 R=3
4120 GOSUB 4480
4130 GOTO 4230
4140 IF CLEF<>3 THEN 4230
4150 FOR I=11 TO 15
4160 CALL HCHAR(I,8,103,19)
4170 NEXT I
4180 CALL HCHAR(16,7,95,20)
4190 CALL VCHAR(7,7,98,8)
4200 CALL HCHAR(15,7,95)
4210 R=11
4220 GOSUB 4480
4230 FOR I=7 TO 13
4240 CALL HCHAR(I,2,136,3)
4250 CALL HCHAR(I,29,136,3)
4260 NEXT I
4270 CALL HCHAR(8,3,128)
4280 CALL HCHAR(8,30,128)
```

```
4290 CALL HCHAR(10,3,137)
4300 CALL HCHAR(10,30,137)
4310 CALL HCHAR(12,3,137)
4320 CALL HCHAR(12,30,137)
4330 RETURN
4340 CALL HCHAR(2,9,96)
4350 CALL HCHAR(2,10,97)
4360 CALL HCHAR(3,9,99)
4370 CALL HCHAR(3,10,100)
4380 CALL HCHAR(4,8,101)
4390 CALL HCHAR(4,9,102)
4400 CALL HCHAR(5,8,104)
4410 CALL HCHAR(5,9,105)
4420 CALL HCHAR(5,10,106)
4430 CALL HCHAR(6,8,107)
4440 CALL HCHAR(6,9,108)
4450 CALL HCHAR(6,10,109)
4460 CALL HCHAR(7,9,99)
4470 RETURN
4480 CALL HCHAR(R,8,110)
4490 CALL HCHAR(R,9,111)
4500 CALL HCHAR(R,10,112)
4510 CALL HCHAR(R+1,8,113)
4520 CALL HCHAR(R+1,10,114)
4530 CALL HCHAR(R+2,9,115)
4540 CALL HCHAR(R+2,10,116)
4550 CALL HCHAR(R+3,8,117)
4560 CALL HCHAR(R+3,9,118)
4570 RETURN
4580 CALL HCHAR(20,3,95,12)
4590 CALL HCHAR(20,19,95,12)
4600 FOR I=0 TO LEN(MSG$)-1
4610 CALL HCHAR(ROW,COL+I,ASC(SEG$(MSG$
     ,I+1,1)))
4620 NEXT I
4630 RETURN
4640 IF LEN(MSG$)=2 THEN 4660
4650 MSG$="_"&MSG$
4660 FOR I=1 TO 2
4670 CALL HCHAR(3,C,ASC(SEG$(MSG$,I,1))
     )
4680 C=C+1
4690 NEXT I
4700 RETURN
4710 FOR MMM=1 TO M
4720 CALL HCHAR(R,6,97)
4730 CALL HCHAR(R-1,6,96)
4740 CALL SOUND(50,-7,5)
4750 CALL SOUND(MM-100,110,30)
4760 CALL HCHAR(R-2,8,97)
4770 CALL HCHAR(R-3,8,96)
4780 CALL SOUND(50,-7,5)
4790 CALL SOUND(MM-100,110,30)
4800 R=R-4
4810 NEXT MMM
4820 CALL SOUND(200,110,30)
4830 RETURN
4840 CALL SOUND(675,262,1)
4850 CALL SOUND(225,294,1)
4860 CALL SOUND(225,311,1)
4870 CALL SOUND(225,110,30)
4880 CALL SOUND(225,262,1)
4890 CALL SOUND(225,110,30)
4900 RETURN
4910 FOR J=12 TO 20 STEP 2
4920 CALL HCHAR(J,2,119,30)
4930 NEXT J
4940 FOR I=F1 TO F1+8 STEP 2
4950 READ MSG$
4960 FOR J=0 TO LEN(MSG$)-1
4970 CALL HCHAR(S1-I,I+J,ASC(SEG$(MSG$,
     J+1,1)))
4980 NEXT J
4990 NEXT I
5000 FOR I=F2 TO F2+6 STEP 2
5010 READ MSG$
5020 FOR J=0 TO LEN(MSG$)-1
5030 CALL HCHAR(S2-I,I+J,ASC(SEG$(MSG$,
     I+1,1)))
5040 NEXT J
5050 NEXT I
5060 MSG$="PRESS ANY KEY"
5070 GOSUB 4600
5080 CALL KEY(0,K,S)
5090 IF S=0 THEN 5080
5100 RETURN
5110 DATA " E - EVERY ",," G - GOOD ",,"
     B - BOY ",," D - DOES ",," F - FINE
     "
5120 DATA F,A,C,E
5130 DATA " G - GOOD ",," B - BOYS ",," D
     - DO ",," F - FINE ",," A - ALWAYS
     "
5140 DATA " A - ALL ",," C - COWS ",," E
     - EAT ",," G - GRASS "
5150 FOR I=3 TO 30
5160 CALL VCHAR(1,I,32,24)
5170 NEXT I
5180 RETURN
```



Notes on a Computer Score: Part 2 - The TI-99/4 Assists Gifted Children in the Learning Process

Although the TI-99/4 proved itself a valuable enrichment tool in my traditional music classes, I began to realize its full potential during a summer program for gifted children at New Horizons Academy. It was exciting to let a curriculum evolve as children enthusiastically identified their own interests and pursued ways of expressing them creatively through use of the computer.

The Educational Setting

New Horizons Academy is a private school that was founded by Nanci Lucas in the belief that children are naturally excited about learning and capable of handling academic pursuits beyond their years. When an individualized curriculum is designed to allow for advancement through basic skills and extensive opportunity for enrichment and acceleration, children find learning to be exciting and meaningful.

In addition to the regular academic curriculum, the Academy periodically provides workshops that are open to any interested children. Since 1978, "Summer Spectacular" has offered courses in computers, creative dramatics, archaeology and photography. I became involved with New Horizons last summer when I taught two sessions in "Computer Music" with our TI-99/4.

The Computer Music classes were intended to familiarize students with basic music concepts and provide for individualized and accelerated learning in a manner consistent with the philosophy of the Academy. A typical group consisted of eight students ranging in age from 7-13. The group was scheduled to meet eight times in a two week period with each session lasting one hour. After the first day, however, the students "demanded" that I arrive at 8:45 A.M. and not leave until 2:15 P.M. Several times it was even 3:45! To see students asking to cut other classes, skipping lunch, and having their parents pick them up late—all just so they could work on their projects—was tremendously rewarding.

The Educational Mode

I employed Renzulli's theoretical framework, *The Enrichment Triad Model*.[1] The model contains three constructs which convey the types of learning activity believed to be best for gifted children.

In *General Exploratory Activities* (Type I) students are exposed to a broad range of possibilities. None of these is presented in detail. The purpose is merely to introduce the students to the range of possible alternatives open to them. *Group Training Activities* (Type II) follow, providing the students with fundamental information of potential use in subsequent development of their interest areas. These activities are oriented toward content.

During the two preceding phases, students begin to identify their interests and develop the skills to create a final product. In *Individual and Small Group Investigations of Real Problems* (Type III), each student determines a problem or project of particular interest that is based on the information obtained in the previous activities, and then pursues that choice in greater depth.

This final element of the Enrichment Triad is perhaps the most important. Ideally, the students will exemplify the "turned-on professional" and pursue their objectives with intense motivation and commitment.

In many respects, Renzulli's model parallels Seymour Papert's principles of continuity, power, and cultural resonance. [2]

Implementation of the Model

In the first few days, I exposed the students to a variety of musical activities including a TI-99/4 concert of familiar children's songs such as "Happy Birthday," "Yankee Doodle," and "Pop Goes the Weasel"—all complete with graphics. We also played rhythm instruments, the autoharp, resonator bells and recorders, drew on impressions of music while listening to recordings, identified environmental sounds, and discussed the commonalities and differences of all sounds.

The Texas Instruments *Music Maker* Command Cartridge contains two options with which children can write music. In the exploratory activities, they utilized the Sound Graphs option, in which "the composer" need not have any prior knowledge of music notation and theory. In this mode the students experimented with duration of notes by con-

trolling the length of the line in whatever voice they were composing (3 voices or 3-part harmony is possible). Frequency is determined by the height of the line on the screen, and there is a volume choice. From all of our exploratory activities, students came to the conclusion that all sounds have duration, frequency, and volume in common. These concepts were effectively and concretely exemplified by the *Music Maker*'s Sound Graphs Mode.

In summary, I believe that the exploratory activities altered the students' *experience* of music. They began to see music in a new way, as part of the continuum of sound and noise; the "freshnesss" of this new perspective contributed to their desire to move toward the next phases of the model.

Group Training Activities were concerned with content-oriented learning. The objectives were to provide students with a basic knowledge of music theory and an understanding of how a computer program is written—information which they could use as tools in developing their interest areas. Several computer music games and drills were used by the entire group, but as a child's interest waned, he was allowed to break away from the group activity and pursue individual work in his primary interest area.

The computer games included (1) *Mystery Words*, in which the players learned the names of treble and bass clef notes; (2) *Rhythm*, which provided ear training in the recognition of quarter notes, eighth notes, and quarter rest patterns; and (3) TI's *Music Skills Trainer*, which contains four games to improve the player's skills in recognition and recall of pitches, intervals, chords, and phrases played by the computer.

We took a look at the program listings for our favorite songs and games and "brainstormed" about what all those commands could possibly mean. Discovering how changing the duration, frequency, and/or volume in a CALL SOUND statement affects the tones produced by the computer was a popular Group Training Activity. The children soon started drawing conclusions and generalizations about how to program. At this point it was necessary to hand out information from the User's Manual for students to take home and study over the weekend—homework at their request!

Additional content-oriented learning took place when students experimented with the Traditional Mode of the computer's *Music Maker* Command Cartridge. I used the traditional mode to help children discover information about key signatures, time signatures, tempo, and music notation—including how various notes such as whole, half, quarter, eighth and sixteenth and their corresponding rests relate to each other and can be organized into a composition that is musically correct.

In moving into Type III of Renzulli's model, almost all students elected to write a computer program to play a musical composition; some students selected pieces with which they were already familar, and others wrote original compositions. Many investigated how to use graphics and color to enhance their creations, and designed title screens to be displayed during the computer's performances of their works. Compositions included "The Entertainer," "Mr. Tambourine Man," "Amazing Grace," Beethoven's Ninth Symphony (*Ode to Joy*), and "Jingle Bells." Three students, Byron, Allan, and Peggy, exhibited a competitive spirit when comparing the number of lines and difficulty of their programs. Steve wrote his program to play Beethoven's Ninth Symphony in three-part harmony; Bryan wrote his original composition to flash a change of screen color to emphasize musical contrasts at appropriate points in the music; Peggy reworked her original composition many times until she was satisfied with the rhythmic structure .

It is important to note that not every student was equally enthusiastic about programming. For example, Adrienne seemed to prefer taking Computer Music for enjoyment and the personal satisfaction of becoming familiar with it but did not have a genuine interest in becoming a creative producer.

"Seeing your students eagerly asking to cut other classes . . . and having their parents pick them up late — all just so they could work on their projects — was tremendously rewarding."

The satisfaction children feel from communicating the results of their work to an audience was obvious when three ladies from Springfield (Ohio) School District and a banker visited our classroom one day. Byron stopped his work to take over for Mrs. Lucas and me as we were explaining the Computer Music Class. He and two others enthusiastically gave a presentation of their music and proudly explained what had gone into its composition. In addition, the already high level of enthusiam increased when the class found out they could present their finished products to their parents and others on Visitation Day and possibly have them published in *99'er Magazine*.

With this group, I served mainly as a resource person and passed the responsibility for learning and investigating on to the students. Students were introduced to concepts of programming and music theory as they explored and used this knowledge to make their programs more complex. It was a good example of making material relevant. The Computer Music course allowed for freedom of choice in that no course requirements were established ahead of time; instead,

the class members were allowed to devolop their own "courses of study" as their interests developed. Likewise, the time allocations were flexible because the entire staff allowed students to skip their classes and come to Computer Music all day if they wanted.

By requiring students to play the games described in Type II activities only until they no longer were interested, mastery of competencies became more streamlined and exciting—as Renzulli suggests. After playing *Mystery Words* about ten minutes, seven-year-old Michael put it this way, "Do we have to play it anymore? We know this now!"

It was interesting to observe how the gifted children mastered the basics much more rapidly and efficiently than my regular general music students. The need for individualization and enrichment for the gifted is obvious when one realizes that playing the same games which intrigued my regular classes for several of their thirty minute music periods tired the gifted in ten to fifteen minutes; they then requested permission to return to programming their *own* creations. This is an example of Renzulli's differentiation of "real investigative activities" for the gifted from "training exercises." This differentiation prompted the development of his Enrichment Triad Model.

It is important to point out that Renzulli's model is not a fixed, rigid framework; the three activity types often overlap. For example, while the actual composing of music is Type III (Individual and Small Group Investigations), the trial-and-error initial discoveries of the computer's musical capabilities might be considered Type I (Exploratory). Likewise, there is overlap in some of the students' other Type II activities. Since brainstorming provides children with the skills needed to explore alternative solutions to problems, our discussions about environmental sounds and computer language were exercises in developing the processes that enable a learner to deal more effectively with content, yet took place during an exploratory activity.

Furthermore, process training (Type II) occurred when students wrote their own programs. As Byron observed, "The computer really programs you; you don't program it." He was referring to the fact that the best programmers learn to think like the computer, in that they think out the process of writing their programs instead of memorizing what to write. Essentially, they must consider how the computer thinks, then determine what to say to get the computer to accomplish their goals. Obviously, the programming required to achieve the final product is a Type III activity, yet the development of thinking processes involved in Type II is also present.

The experience of teaching at New Horizons has given me new insights into how children think and how different their learning styles can be. It was exciting to allow a curriculum to evolve as children enthusiastically identified their own interests and pursued ways of expressing these interests creatively.

Postscript

Although the Academy already had four CBM computers and a TRS-80, the magical attraction of students to our TI-99/4 did not go unrecognized. Soon the Academy purchased a TI-99/4A, and subsequently a second one, together with a variety of the high quality educational software offered by Texas Instruments.

My husband and I have conducted several other enrichment sessions at the Academy and have become increasingly excited about the profound potential of computer-facilitated learning.

References

1 Renzulli, Joseph S. *The Enrichment Triad Model: A Guide for Developing Defensible Programs for the Gifted and Talented.* Connecticut: Creative Learning Press, Inc., 1977.

2 Papert, Seymour. *Mindstorms: Children, Computers, and Powerful Ideas.* New York: Basic Books, Inc., 1980.

A Music Text Editor & File Player for the TI-99/4A

TI BASIC

For those readers who do not as yet have a *Music Maker* but would like to experiment with music writing anyway, we are offering a primitive music text editor, which has two of the three voices of the TI99/4A, as well as a file player program and the input required to play.

The *Music Text Editor* program creates a tape file which is read and played by the *Music File Player* that follows it. Although the file can be played by the editor, the tempo will be somewhat slower than when performed by the separate player program.

Use the following symbols to write note values:

W-Whole	H-Half
Q-Quarter	E-Eighth
S-Sixteenth	

For a dotted note value, put a period after the symbols, for example, S.,Q., etc.

Use the following symbols for pitches:

A A#	E
B	F F#
C C#	G G#
D D#	R-Rest (a pitch with value 0)

After each pitch, give the octave (1-4):

Octave	Begins
1	Bass clef, bottom space (A = 110 cps)
2	Bass clef, top line (A = 220 cps)
3	Treble clef, 2nd space from bottom (A = 440 cps)
4	Treble clef, 1st ledger line above (A = 880 cps)

[**NOTE:** *SAVE your input periodically! An input of any note plus an accidental but* ***without*** *an octave number (e.g., A#) may cause your program to crash.]*

The *Music Text Editor* will first ask you for the composition's and the composer's names. Then the prompt M.M will ask you for a tempo in quarter notes per minute—corresponding approximately to a metronome beat—between 56 and 126 per minute. (NOTE: The program won't accept any value larger than 126; the computer will reject an overly-funereal beat because the value put in the duration parameter of CALL SOUND will be too large.)

The program then begins requesting input for the composition, line by line for the two voices. After the program displays the line number, you enter the duration (W, Q., S, . . .), followed by the pitch values and octave ranges for each of the two voices. You must separate these values by a slash (/), and end the line with a slash. For instance, to play simultaneously the dotted eighth notes **F ♯** in octave 2 and **C** in octave 1, followed by a sixteenth rest in the first voice and a **B♭** sixteenth note in the second, after the first program line number prompts (1 =) you would enter:

```
1 = E./F#2/C1/
2 = S/R/A#1/
3 =
```

When the program prompts you for the next line of notes (in this instance line 3), you may instead enter one of the following commands:

CHANGE, REDO, LIST, PLAY, or SAVE.

CHANGE, REDO and LIST will ask for a range of lines. The SAVE command merges new data with data already stored in the tape file. Unless you answer the question, "FINISHED?" with a "YES", no end-of-file mark will be written on the tape file. Until the file has an end-of-file mark, it can't be read. The *Music File Player* can read and play a file consisting of up to 550 lines.

```
120 REM  *  MUSIC TEXT EDITOR  *
160 REM
170 REM
180 CALL CLEAR
190 DIM N$(11,4),V$(8,1),P(349,2)
200 INPUT "TITLE      - ":T$
210 INPUT "COMPOSER   - ":C$
220 FOR I=0 TO 11
230 READ N$(I,0)
240 NEXT I
250 DATA A,A#,B,C,C#,D,D#,E,F,F#,G,G#
260 FOR I=1 TO 4
270 FOR J=0 TO 11
280 X=110*2^(S/12)
290 GOSUB 1420
300 N$(J,I)=STR$(X)
310 S=S+1
320 NEXT J
330 NEXT I
340 CALL CLEAR
350 PRINT "ENTER TEMPO (M.M.)":"QUARTE
    R NOTES/MINUTE"::
360 INPUT "M.M = ":T
370 IF T<127 THEN 410
380 PRINT :"MAXIMUM IS 126"::
390 CALL SOUND(150,220,1)
400 GOTO 360
410 CALL CLEAR
420 T=6E+4/T
430 FOR I=0 TO 8
440 FOR J=0 TO 1
450 READ V$(I,J)
460 NEXT J
```

```
470 NEXT I
480 DATA W,4,H,2,H.,3,Q,1.1,Q.,1.5,E,.
5,E.,.75,S,.25,S.,.375
490 FOR I=0 TO 8
500 X=VAL(V$(I,1))
510 X=T*X
520 GOSUB 1420
530 V$(I,1)=STR$(X)
540 NEXT I
550 CALL CLEAR
560 KP=0
570 GOSUB 1330
580 PO=1
590 X$=""
600 FOR N=1 TO 4-LEN(STR$(K+KP+1))
610 X$=X$&" "
620 NEXT N
630 INPUT STR$(K+KP+1)&X$&" = ":X$
640 IF K=350 THEN 670
650 P(K,1)=111
660 P(K,2)=111
670 IF X$<>"CHANGE" THEN 700
680 GOSUB 1570
690 GOTO 580
700 IF X$<>"PLAY" THEN 730
710 GOSUB 1200
720 GOTO 580
730 IF X$<>"REDO" THEN 760
740 GOSUB 1470
750 GOTO 580
760 IF X$<>"LIST" THEN 790
770 GOSUB 1710
780 GOTO 580
790 IF X$<>"SAVE" THEN 810
800 GOSUB 2160
810 IF K<>350 THEN 840
820 PRINT "SAVE FILE BEFORE PROCEEDING
"::
830 GOTO 580
840 GOSUB 1370
850 IF S$="" THEN 870
860 IF PN<>0 THEN 890
870 CALL SOUND(150,220,1)
880 GOTO 580
890 FOR I=0 TO 8
900 IF S$=V$(I,0)THEN 950
910 NEXT I
920 PRINT "ERR- VALUE"
930 CALL SOUND(150,220,1)
940 GOTO 580
950 P(K,0)=VAL(V$(I,1))
960 FOR I=1 TO 2
970 GOSUB 1370
980 IF S$="" THEN 1000
990 IF PN<>0 THEN 1020
1000 CALL SOUND(150,220,1)
1010 GOTO 580
1020 IF S$="R" THEN 1150
1030 IF SEG$(S$,LEN(S$),1)<"5" THEN 106
0
1040 CALL SOUND(150,220,1)
1050 GOTO 580
1060 OC=VAL(SEG$(S$,LEN(S$),1))
1070 S$=SEG$(S$,1,LEN(S$)-1)
1080 FOR J=0 TO 11
1090 IF S$=N$(J,0)THEN 1140
1100 NEXT J
1110 PRINT "ERR- VOICE";I
1120 CALL SOUND(150,220,1)
1130 GOTO 580
1140 P(K,I)=VAL(N$(J,OC))
1150 NEXT I
1160 IF SR=0 THEN 1180
1170 RETURN
1180 K=K+1
1190 GOTO 580
1200 FOR L=0 TO K-1
1210 IF P(L,1)<>111 THEN 1270
1220 IF P(L,2)<>111 THEN 1250
1230 CALL SOUND(P(L,0),110,30,110,30)
1240 GOTO 1310
1250 CALL SOUND(P(L,0),110,30,P(L,2)-1)
1260 GOTO 1310
1270 IF P(L,2)<>111 THEN 1300
1280 CALL SOUND(P(L,0),P(L,1),1,110,30)
1290 GOTO 1310
1300 CALL SOUND(P(L,0),P(L,1),1,P(L,2),
1)
1310 NEXT L
1320 RETURN
1330 FOR I=0 TO 349
1340 P(I,0)=0
1350 NEXT I
1360 RETURN
1370 PN=POS(X$,"/",PO)
1380 IF PN=0 THEN 1410
1390 S$=SEG$(X$,PO,PN-PO)
1400 PO=PN+1
1410 RETURN
1420 Y=X-INT(X)
1430 IF Y<5 THEN 1450
1440 X=X+1
1450 X=INT(X)
1460 RETURN
1470 M=K
1480 INPUT "START AT LINE - ":K
1490 IF K>KP THEN 1520
1500 CALL SOUND(150,220,1)
1510 GOTO 1480
1520 K=K-KP-1
1530 IF K<M+1 THEN 1560
1540 CALL SOUND(150,220,1)
1550 GOTO 1480
1560 RETURN
1570 SR=1
1580 M=K
1590 INPUT "CHANGE LINE - ":K
1600 IF K>KP THEN 1630
1610 CALL SOUND(150,220,1)
1620 GOTO 1590
1630 K=K-KP-1
1640 IF K<M THEN 1670
1650 CALL SOUND(150,220,1)
1660 GOTO 1590
1670 GOSUB 580
1680 K=M
1690 SR=0
1700 RETURN
1710 INPUT "FIRST LINE - ":O
1720 IF O>KP THEN 1750
1730 CALL SOUND(150,220,1)
1740 GOTO 1710
1750 INPUT "LAST LINE - ":Q
1760 IF Q>KP THEN 1790
1770 CALL SOUND(150,220,1)
1780 GOTO 1750
1790 IF O<=Q THEN 1820
1800 CALL SOUND(150,220,1)
1810 GOTO 1710
1820 IF Q-KP<K+1 THEN 1840
1830 Q=K+KP
1840 PRINT ::
1850 O=O-KP-1
1860 Q=Q-KP-1
1870 FOR R=O TO Q
1880 X$=""
1890 FOR I=1 TO 4-LEN(STR$(R+1))
1900 X$=X$&" "
1910 NEXT I
1920 PRINT STR$(R+1)&X$&" - ";
1930 FOR S=0 TO 8
1940 IF P(R,0)=VAL(V$(S,1))THEN 1960
1950 NEXT S
1960 PRINT V$(S,0)&"/";
1970 FOR I=1 TO 2
1980 IF P(R,I)<>111 THEN 2040
1990 IF I<>2 THEN 2020
2000 PRINT "R/"
```

```
2010 GOTO 2140
2020 PRINT "R/";
2030 GOTO 2120
2040 FOR I1=0 TO 11
2050 FOR I2=1 TO 4
2060 IF P(R,I)<VAL(N$(I1,I2))THEN 2090
2070 IF P(R,I)=VAL(N$(I1,I2))THEN 2100
2080 NEXT I2
2090 NEXT I1
2100 IF I=2 THEN 2130
2110 PRINT N$(I1,0)&STR$(I2)&"/";
2120 NEXT I
2130 PRINT N$(I1,0)&STR$(I2)&"/"
2140 NEXT R
2150 RETURN
2160 IF F$="1" THEN 2200
2170 F$="1"
2180 OPEN #1:"CS1",SEQUENTIAL,INTERNAL,
     OUTPUT,FIXED 192
2190 PRINT #1:T$;C$
2200 I1=0
2210 K=K-1
2220 FOR I=1 TO 12
2230 FOR J=0 TO 2
2240 PRINT #1:STR$(P(I1,J)),
2250 NEXT J
2260 I1=I1+1
2270 IF I1>K THEN 2330
2280 NEXT I
2290 PRINT #1:STR$(P(I1,0)),STR$(P(I1,1
     )),STR$(P(I1,2))
2300 I1=I1+1
2310 IF I1>K THEN 2340
2320 GOTO 2220
2330 PRINT #1:""
2340 PRINT "FINISHED? (Y/N)"
2350 CALL KEY(0,KEY,STATUS)
2360 IF KEY=89 THEN 2410
2370 IF KEY<>78 THEN 2350
2380 KP=KP+K+1
2390 K=0
2400 GOTO 570
2410 PRINT #1:"@@@@@"
2420 CLOSE #1
2430 PRINT T$&" SAVED"
2440 END
```

```
 120 REM * MUSIC FILE PLAYER *
 160 REM
 170 REM
 180 CALL CLEAR
 190 PRINT "THIS PROGRAM PLAYS A MUSIC
     FILE CREATED BY THE 2-VOICE EDITO
     R."
 200 PRINT :"* PLACE DATA IN CASSETTE C
     S1":"THEN PRESS ENTER"
 210 CALL SOUND(150,1400,1)
 220 CALL KEY(0,KEY,STATUS)
 230 IF KEY<>13 THEN 220
 240 DIM P(530,2)
 250 OPEN #1:"CS1",SEQUENTIAL,INTERNAL,
     INPUT,FIXED 192
 260 PRINT :"* READING"
 270 CALL SOUND(150,1400,1)
 280 INPUT #1:N$,C$
 290 FOR I=0 TO 530
 300 FOR J=0 TO 2
 310 INPUT #1:X$,
 320 IF X$="" THEN 310
 330 IF X$="@@@@@" THEN 370
 340 P(I,J)=VAL(X$)
 350 NEXT J
 360 NEXT I
 370 CALL CLEAR
 380 PRINT TAB((28-LEN(N$))/2);N$::TAB(
     13);"BY"::TAB((28-LEN(C$))/2);C$::
     ::::::::
 390 FOR K=0 TO I-1
 400 IF P(K,1)<>111 THEN 440
 410 IF P(K,2)<>111 THEN 470
 420 CALL SOUND(P(K,0),110,30,110,30)
 430 GOTO 500
 440 IF P(K,2)<>111 THEN 490
 450 CALL SOUND(P(K,0),P(K,1),1,110,30)
 460 GOTO 500
 470 CALL SOUND(P(K,0),110,30,P(K,2),1)
 480 GOTO 500
 490 CALL SOUND(P(K,0),P(K,1),1,P(K,2),
     1)
 500 NEXT K
 510 CALL CLEAR
 520 PRINT "SELECT FROM...":::"(1) PLAY
     IT AGAIN":"(2) NEXT PIECE":"(3)
     STOP"
 530 CALL KEY(0,KEY,STATUS)
 540 IF KEY<49 THEN 530
 550 IF KEY>51 THEN 530
 560 KEY=KEY-48
 570 ON KEY GOTO 370,280,580
 580 CLOSE #1
 590 STOP
```

Fugue in G minor

MM = 100 by Bach

1 = Q/G2/R/
2 = Q/D3/R/
3 = Q/A#3/R/
4 = E/A3/R/
5 = E/G2/R/
6 = E/A#3/R/
7 = E/A3/R/
8 = E/G2/R/
9 = E/F#2/R/
10 = E/A3/R/
11 = Q/D2/R/
12 = E/G2/R/
13 = E/D2/R/
14 = E/A3/R/
15 = E/D2/R/
16 = E/A#3/R/
17 = S/A3/R/
18 = S/G2/R/
19 = E/A3/R/
20 = E/D2/R/
21 = E/G2/R/
22 = S/D2/R/
23 = S/G2/R/
24 = E/A3/R/
25 = S/D2/R/
26 = S/A3/R/
27 = E/A#3/R/
28 = S/A3/R/
29 = S/G2/R/
30 = S/A3/R/
31 = S/D2/R/
32 = S/D3/R/
33 = S/C3/R/
34 = S/A#3/R/
35 = S/A3/R/
36 = S/G2/R/
37 = S/A#3/R/
38 = S/A3/R/
39 = S/G2/R/
40 = S/F#2/R/
41 = S/A3/R/
42 = S/G2/R/
43 = S/D2/R/
44 = S/G2/R/
45 = S/A3/R/
46 = S/A#3/R/
47 = S/C3/R/
48 = S/D3/R/
49 = S/E3/R/
50 = S/F3/D2/
51 = S/E3/D2/
52 = S/D3/R/
53 = S/F3/R/
54 = S/E3/A3/
55 = S/D3/A3/
56 = S/C#3/R/
57 = S/E3/R/
58 = S/D3/F2/
59 = E/A3/F2/
60 = E/D3/F2/
61 = S/E3/E2/
62 = S/E3/E2/
63 = S/F3/D2/
64 = S/G3/D2/
65 = S/F3/F2/
66 = S/G3/F2/
67 = E/G3/E2/
68 = E/G3/E2/
69 = E/G3/D2/
70 = S/F3/D2/
71 = S/G3/D2/
72 = S/A4/C#2/
73 = S/G3/C#2/
74 = S/A4/C#2/
75 = S/A#4/E2/
76 = S/A4/E2/
77 = S/G3/E2/
78 = S/F3/A2/
79 = S/E3/A2/
80 = S/F3/D2/
81 = S/A4/A2/
82 = S/G3/A2/
83 = S/A4/A2/
84 = S/C#3/E2/
85 = S/A4/E2/
86 = S/G3/A2/
87 = S/A4/A2/
88 = S/D3/F2/
89 = S/A4/F2/
90 = S/G3/E2/
91 = S/A4/D2/
92 = S/C#3/E2/
93 = S/A4/E2/
94 = S/G3/E2/
95 = S/A4/A2/
96 = S/F3/D2/
97 = S/D3/D2/
98 = S/C#3/A2/
99 = S/D3/D2/
100 = S/G3/E2/
101 = S/D3/A2/
102 = S/C#3/A2/
103 = S/D3/E2/
104 = S/A4/F2/
105 = S/D3/F2/
106 = S/C#3/E2/
107 = S/D3/D2/
108 = S/G3/E2/
109 = S/D3/A2/
110 = S/C#3/A3/
111 = S/D3/G2/
112 = E/A3/F2/
113 = E/F3/D2/
114 = E/G2/E2/
115 = E/E3/C#2/
116 = E/F2/D2/
117 = E/A3/D2/
118 = E/D3/F2/
119 = S/F3/A3/
120 = S/F3/B3/
121 = E/D#3/C3/
122 = E/A4/C3/
123 = E/A4/D3/
124 = E/D#3/A#3/
125 = S/D3/C3/
126 = E/D3/A#3/
127 = E/G3/A#3/
128 = E/G3/A#3/
129 = S/D3/A3/
130 = E/D3/G2/
131 = S/R/A#3/
132 = S/C3/A3/
133 = S/A#3/A3/
134 = S/C3/G2/
135 = S/D3/G2/
136 = S/C3/F#2/
137 = S/A4/F#2/
138 = S/G3/D2/
139 = S/A4/D2/
140 = S/A#3/G2/
141 = S/G3/R/
142 = S/F#3/R/
143 = S/G3/R/
144 = S/A3/D2/
145 = S/F3/R/
146 = S/E3/R/
147 = S/F3/R/
148 = E/G1/A#2/
149 = E/D#3/R/
150 = E/G2/R/
151 = E/A2/R/
152 = E/A#2/E1/
153 = E/R/D1/
154 = E/R/F1
155 = E/R/G1
156 = Q/D1/R/
157 = Q/A2/R/
158 = Q/F1/R/
159 = E/E1/R/
160 = E/D1/R/
161 = E/F1/R/
162 = E/E1/R/
163 = E/D1/R/
164 = E/C#1/A2/
165 = E/E1/A#2/
166 = Q/A1/R/
167 = S/R/F2/
168 = S/R/E2/
169 = E/D1/F2/
170 = E/A1/R/
171 = E/E1/C#2/
172 = E/F1/D2/
173 = Q/A1/D2/

Music Maker

A TI Command Cartridge

To paraphrase Shakespeare, "The computer that hath no music in its chips, nor is not programm'd with concord of sweet sound is fit but for business, mathematics, and sorts." The TI-99/4A is definitely not one of these.

Outstanding music and sound effects capabilities are among the many features which set the TI-99/4A apart from other personal computers. A user can generate three simultaneous tones and a noise, and can specify their duration, pitch, and loudness—all with a single TI BASIC statement. The sound is played through the speaker of the color monitor or TV display.

Of course, an assortment of beeps,"ta-daas" and outer space sounds can greatly enhance a graphics presentation and provide useful auditory feedback during the program execution. But when the sophisticated sound capabilities of the TI-99/4A become the focus of the programmer's attention, the Texas Instruments machine becomes a musical instrument in its own right. Whether playing a Bach sonata or your own composition, a successful TI-99/4A performance is worth the programming effort.

With the introduction of TI's *Music Maker* Command Cartridge, you can take full advantage of the TI-99/4A's sound capabilities without having to write a complex BASIC program. The *Music Maker* allows you to write a composition using either of two methods—Traditional Mode or Sound Graphs. While Traditional Mode requires some knowledge of fundamental music theory, Sound Graphs does not. Both methods are graphics-based, in contrast to other music editor formats which require entry of notes using ASCII characters. Both also make superb use of the TI-99/4A's outstanding color graphics capabilities. Notes are entered by manipulating the cursor with either the joysticks or the arrow keys. A composer can then print out the bass and treble clefs of each measure—complete with all notes, sharps, flats, and rests—with TI's thermal printer (using its special graphic character set). It's also possible to save the completed musical score on cassette tape or diskette.

Traditional Mode

In Traditional Mode, notes are entered directly on the music staff using standard notation. The first step involves defining the key, meter, and tempo. All possible key signatures (0-7 sharps or flats) are allowed. The meter or time signature options for the denominator are 1, 2, 4, 8, and 16—corresponding to the unit of measure receiving one beat (i.e., whole, half, quarter, eighth or sixteenth note). The numerator of the time signature indicates the number of such units which comprise a measure. Your options here are restricted to values equal to, or less than, the denominator. Examples of allowable time signatures are:

$$\frac{4}{4}, \frac{6}{8}, \frac{2}{2}, \text{ and } \frac{3}{4};$$

on the other hand,

$$\frac{3}{2}, \frac{12}{8}, \text{ and } \frac{5}{4}$$

are not allowable, because the numerator exceeds the

denominator. This limitation is significant, because there is a natural accent which falls on the first beat of every measure when music is accurately interpreted by a performer. This regular impulse, together with phrasing and secondary accents in compound meter, gives a composition its underlying rhythmic structure. The *Music Maker* does not automatically provide for this natural rhythm. The implementation of accent is entirely up to you. For example, a composition written in 4/4 time may be made to sound like 3/2 time with proper phrasing and specification of accent. Therefore, the time signature limitation does not actually limit the music you can write with the cartridge. Finally tempo is specified as a number from 1 to 30, corresponding approximately to metronomic indications from 25 to 128 quarter notes per minute—sufficient range for nearly all compositions.

After these parameters are defined, the graphics representation for the first measure appears. Some music editors for other machines do not use graphics at all. It is a great advantage to see your composition displayed in standard notation as you are writing it.

Up to three voices may be "drawn" using whole, half, quarter, eighth and sixteenth notes and their corresponding rests. Single dotting can be used with notes, but not with rests. The notes for each voice are represented in a different color, which facilitates identification of voices when editing.

The pitch range is three octaves, extending from the second **A** below middle **c** (bottom space of bass clef) to the second **a** above middle **c** (first ledger line above treble clef). This may seem like a wide range. In arranging several piano pieces for the TI-99/4A, however, we found that it was frequently necessary to make octave transpositions for notes extending beyond the *Music Maker's* pitch range in Traditional Mode. On the other hand, the *Music Maker* is not really intended for the transcription of existing music for other instruments, but rather to facilitate original composition. Like all instruments, it too has limitations which must be taken into account when preparing an original composition.

Accidentals (sharps, flats, or naturals different from the key signature) must be written for each note; once written, they do not carry over through the entire measure as they do in standard notation. For someone who is accustomed to standard notation, this may take a little getting used to. Additionally, the large and legible graphic symbols that the cursor picks up from the menu become too small to be easily read when placed beside a note.

Graphics

Graphics characters for the notes themselves resemble square notation, but we do not feel this detracts from their readability (especially when compared with the legibility of many manuscripts). However, in drawing clusters of two or more notes, we encountered a peculiar graphics-related difficulty. This is a function of the position (up or down) of an existing note stem. You will find that a note for one voice can not be placed at a pitch immediately above or below an existing note if that pitch is occupied by the stem of the existing note. The stems for voices one and two go upward unless they are placed immediately below a note in another voice which has its stem going upward. The opposite is true of voice-three notes. This means that while it is possible to represent any two-note cluster, the process can be more involved than would seem necessary. For instance, suppose you have already written a voice-one quarter note at middle **c**, and you want to write a voice-two note at **d** immediately above it. Finding that you cannot do this simply, you would have to do the following: Change voices, erase the **c**, change voices, draw the **d** (voice-two), change voices, redraw the **c** (stem down), and finally change back to voice-two to continue. A cluster of three notes with adjacent pitches cannot be written at all. These problems will be troublesome only in the event that the composer wants to write dissonant chords in the form of clusters.

At the bottom of the display is a double row of squares; the upper row is used to specify volume for each note. There are eight levels of volume which allow a very smooth crescendo or diminuendo without abrupt transitions from one level to the next. Some other music editors do not allow this degree of versatility in dynamics. This default value for loudness is the maximum level of eight. If you want to accent selected notes, say the first note of every measure, you must drop the volume of all other notes. A default loudness of six or seven might have been a little easier to use in this regard.

The bottom row of squares is used to indicate the width of each note; this is very helpful in positioning them. It also allows one to create rests without using rest graphics by simply leaving a gap between one note and the next. Two adjacent notes of the same pitch are automatically tied. The only way to articulate them is to leave a gap in between. For instance, one might write a dotted quarter rather than a quarter note, and the resulting gap would then prevent a tie with the next note.

At any point during the writing of a measure, you can play an individual voice or all voices. If you decide to make a change, this is easily accomplished by erasing an individual note or the entire voice. You cannot, however, insert or delete notes without making necessary adjustments to other notes in the measure.

Repetition is easily handled by copying an individual voice or all voices from a previous measure, and this can save a great deal of time. A given voice cannot, however, be copied as another voice. So if you want to use the copy feature to write rounds, they have to be scored differently than they would be in traditional composition. Any two voices can be copied by copying all three and then erasing the one which is not wanted.

When you are finished with a measure, you can either go on to the next measure or back to a menu which allows you to edit, play, save, or print your composition. If you choose to edit, you will be shown the number of measures completed and the percentage of file space used, and you will be given the option of changing the tempo. To play the composition, you specify which voices are to be played, and you are given the option of hearing the music transposed up or down by as many as eleven half steps (twelve half steps are an octave). If you transpose a note so that it falls *below* the *Music Maker's* range, it will not be played. You can interrupt the playing of a composition and view the graphic representation of the measure being played at that point, but graphics are not used when the piece is actually being played.

There are a few features present in some music editors for other machines which are not present in *Music Maker*. For example, the only way to initiate repeats is by manually pressing "SHIFT R" during the playing of a piece; no form of looping can be structured into a compostition.

Given the relatively vast storage space available (compared with music compositions written in TI BASIC), together with the copying feature, the lack of repeat capability is less significant than it might otherwise be. With 16K of RAM, you will be able to write about 900 notes for each of the three voices. For example, writing all sixteenth notes for three voices, the file could be 57 measures long; with all quarter notes, it could be 224 measures. Additionally, there is no capability to write phrases and then arrange them in different voices. This capability could be useful when employing the device of imitation, such as writing canons and fugues. Even so, the same effect can be achieved with *Music Maker*—it just takes a little more effort.

In summary, despite the few shortcomings mentioned, the Traditional Mode provides a beautiful graphics-based editor which makes the process of writing music as enjoyable as listening to the finished product. Even if this were the extent of the *Music Maker*'s capabilities, we feel it would be an excellent investment at the suggested retail price of $39.95.

Sound Graphs

While some knowledge of music theory is essential for effective use of the Traditional Mode, the Sound Graphs method may be used without any prior understanding of music terminology. As the name implies, music is entered in a Cartesian coordinate graph format. The frequency graph can have a resolution of one-hundred-twenty vertical positions (frequency) by twenty horizontal positions (duration) per "measure." A Sound Graphs music file may contain up to 46 measures. A color-coded line is plotted on the graph with the cursor, and as in Traditional Mode, a different color is used for each voice.

The volume graph has a resolution of eight vertical positions (volume) by twenty horizontal positions (duration), and appears below the frequency graph. A separate volume graph may be plotted for each voice appearing in the frequency graph (default is the highest volume). In addition to the three voices, a Sound Graph may also include a noise which is plotted on the volume graph.

The user has the option of either Discrete or Continuous tones. Under the Discrete option, the vertical axis is divided into thirty frequencies, consisting of **C** Major diatonic pitches from the second **A** below middle **c** to the third **b** above middle **c**. You can, however change any or all of these pitches with the List Tones option. Although any frequency from 110 Hz to 20,000 Hz can be used, tables are provided in TI's excellent documentation, giving the frequencies for chromatic, pentatonic, and gypsy scales. The frequencies can be changed at any time, even during or after the plotting of a Sound Graph.

Under the Continuous option you specify the upper and lower limits of the frequency range. These can be changed as often as you wish. The frequency axis is divided into 120 steps within this range, giving a frequency "slide" which sounds continuous and can be used to create sound effects such as whistles and sirens as well as interesting experimental music sounds. When you take into consideration the fact that a noise can be used in addition to three voices and that the composition can be played as fast as twenty characters per second, the range of possibilities is quite extensive.

In evaluating the noise, we were surprised to find that we could not distinguish any difference between the periodic and "white noise" groups—i.e., noises 1-4 and 5-8, respectively. Noise 1 appears to be the same as noise 5; noise 2 the same as 6, and so on. If you are familiar with the difference between periodic noise and white noise in TI BASIC, do not expect to find the same distinction in the *Music Maker*.

Other aspects of using Sound Graphs are identical with the corresponding procedures used in Traditional Mode (namely editing, playing, saving, and printing).

If you have no knowledge of music theory, using Sound Graphs is a great way to begin exploring the TI-99/4A's music capabilities. Even if you are familiar with music fundamentals, you will be amazed at the versatility of the Sound Graphs method, and you will find that your TI-99/4A has potential you would not have thought possible.

In conclusion, the *Music Maker* Command Cartridge will greatly enhance one of the already outstanding features of your computer—its capacity for sound and music. We believe it is an accessory you will not want to be without.