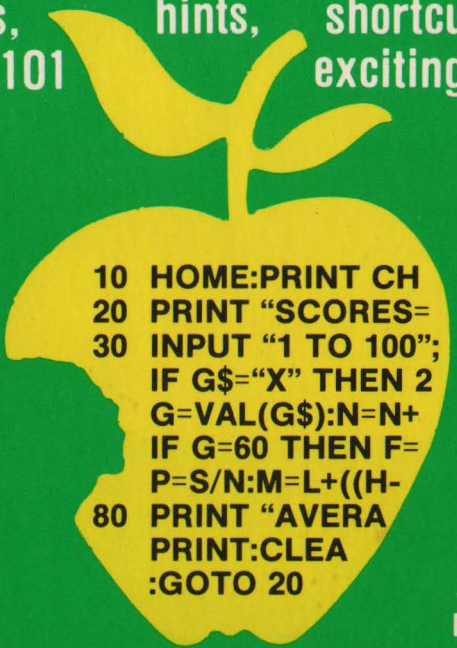
$8[illegible]

101 APPLE COMPUTER PROGRAMMING TIPS & TRICKS

The software ideabook, overflowing with pro techniques, hints, shortcuts, secrets and with 101 exciting programs.



by Fred White

101 APPLE COMPUTER PROGRAMMING TIPS & TRICKS

by Fred White

ARCsoft Publishers

WOODSBORO, MARYLAND

FIRST EDITION
FOURTH PRINTING

ISBN 0-86668-015-2

Preface

The APPLE Computer is one of the world's most popular computer systems. Its lightweight, desktop design and powerful version of the BASIC programming language place it at the forefront of the new wave of personal computers for home, school and office.

Anything but a toy, its hardware configuration and system software make it a highly useful tool in the business environment and the classroom as well as for practical jobs around the home.

In fact, the system software is so flexible and versatile that the need for this book became apparent. There are so many computer tasks which can be accomplished with the APPLE system that an introduction to the many techniques is needed.

Software programs are what make a computer do what you want. This book is written for those newcomers and beginners, as well as advanced novices and student programmers, who would like to tap the vast resources available in the APPLE computer package. This book, it is hoped, will guide and instruct and provide insights into the many ways the BASIC language built into the APPLE computer can be put to use.

This book is a companion volume to *33 New APPLE Computer Programs for Home, School & Office.*

— Fred White

Table of Contents

Text On Text

Gee Whiz

Number Crunching

Business Matters

Introduction

This book is designed for beginners and newcomers, as well as advanced novices and student programmers, who have read a book on how to program in BASIC and now need some good ideas.

This handbook gives you 101 different programs which will stand alone and run by themselves for learning purposes and can be incorporated into larger sets of instructions.

These are designed to be typed in, just as you find them in the book, with no other programming needed. You don't have to be a computer programmer to use these pieces of software.

The book can be used by newcomers and beginners as well as old-timers in the programming game.

Amidst our 101 tips are hundreds of secrets, shortcuts, tricks, hints, techniques and make-it-easier instructions, each designed to make you a more versatile programmer and to make your programming effort lighter. Use this book to stimulate your thinking about how to approach various software problems. Use it to get good ideas for

new and different approaches to all your programming goals.

Very few REMs

As you read through the 101 programs in this book, you will notice very little use of REMarks. The author's training in writing BASIC-language computer programs required a sharp editing pencil. Remarks and software explanations were out. Honing, fine-tuning and waste-trimming were in. Use of coding-form programming worksheets, such as the *Universal BASIC Coding Form,* published by ARCsoft Publishers, was in. The objective always was—and still is—to make the most efficient use of available memory.

Even though they may be headed toward the same goal, no two programmers will write exactly the same list of BASIC program lines from scratch. I'm sure that as you load these into your Computer, you'll make modifications to suit your personal needs or interests. Exact wording of PRINT statements can be changed. Two and more programs can be combined into one grand scheme. Applications will vary.

The author would like to hear of improvement ideas as well as suggestions for future volumes in this series. He may be contacted in care of ARCsoft Publishers, P.O. Box 132, Woodsboro, MD 21798 USA.

Standalone vs. subroutine

All of the programs in this book can be used as portions of larger listings, as subroutines or GOTO objects. To do so, make appropriate changes to the first line (often numbered 10 in this book) and the last line of each program. If you wish to create a subroutine, remember that RETURN must be the last line of the subroutine. If you work one of these programs into a larger list as a subroutine, be especially careful of your memory (variable) labels. They must agree with your selection of labels for the main program. Also, be careful of line numbering.

If you want to load more than one of these programs into your Computer at the same time, be sure to use different sets of line numbers for different programs.

Other computers

These programs will run on other microcomputer systems but you'll have to make minor modifications to the program lines in some cases. The graphics commands and sound statements, especially, will differ elsewhere.

Fun & Games

1 Super Slot-0

Let's start our Fun-N-Games section with instruction on how to simulate a slot machine on the screen of your APPLE!

As with all the programs used as examples in this book, simply type this one in and RUN it. The APPLE will display, on your video screen, the name of this program and some simple instructions.

To simulate pulling the slot machine's lever arm, press any key on the keyboard of your APPLE.

By the way, be very careful in typing in the program. Note that line 100 is very long. Yes, all that goes in one program line!

Be sure your line 90 has six sets of five asterisks each. Lines 500, 510 and 520 start with nine blank spaces before the first asterisk. Lines 500 and 520 have 16 asterisks each.

It is important that you include the called-for blank spaces in lines 90, 100, 110, 500, 510 and 520. As you key in programs throughout this book, be sure you include all blank spaces where called for.

Program Listing

```
 10 HOME
 20 GOSUB 500
 30 PRINT:PRINT
 40 DATA 1,2,3,4,5,6,1,2,3,4,5,6
 50 FOR LL=1 TO 6
 60 GOSUB 300
 70 C$(LL)=G$
 80 NEXT LL
 90 PRINT "***** ***** ***** *****
    ***** *****"
100 PRINT "* ";C$(1);" * * ";C$(2)
    ;" * * ";C$(3);" * * ";C$(4)
    ;" * * ";C$(5);" * * ";C$(6);"
110 PRINT "***** ***** ***** ** **
    ***** *****"
```

```
120 PRINT:PRINT
130 PRINT"TO PULL THE LEVER, PRESS ANY KEY"
140 GET Q$
150 HOME:GOTO 20
300 FOR L=1 TO 6
310 READ P$(L)
320 NEXT L
330 RESTORE
340 R=INT(7*(RND(1)))
350 IF R<1 THEN 340
360 G$=P$(R)
370 RETURN
500 PRINT "          **********************"
510 PRINT "          * SUPER SLOT-O *"
520 PRINT "          **********************"
530 PRINT CHR$(7)
540 RETURN
```

2 See Two Dice

Here's a quick way to add real dice to any fun program you are designing for your APPLE.

This program rolls two dice and lets you see the results, as with real dice. This is especially useful in those games where it is important to see the value of each.

The subroutine in lines 100-140 generates the necessary pair of random numbers. Lines 60, 70 and 80 make the display you want.

Note that lines 60 and 80 each have nine asterisks. Line 140 is RETURN and must be the last line in the program.

After you type in and RUN the program, press any key on your APPLE's keyboard to roll the dice.

Program Listing

```
10 HOME
20 PRINT "TO ROLL TWO DICE, PRESS ANY
   KEY"
```

```
 30 GET K$
 40 HOME
 50 GOSUB 100
 60 PRINT "**********"
 70 PRINT "* ";DL;" * ";DR;" *"
 80 PRINT "**********"
 90 PRINT:PRINT:GOTO 20
100 DL=INT(7*(RND(1)))
110 IF DL<1 THEN 100
120 DR=INT(7*(RND(1)))
130 IF DR<1 THEN 120
140 RETURN
```

3 See Four Dice

Two dice not enough for your game? Here's how to see four dice after a roll!

Naturally, this program works just like the program in tip number two except that the FOR/NEXT loop in lines 50-110 makes the APPLE roll and display four times rather than two times. If you need six, eight or ten dice on display, change the number two in line 50 to three, four or five.

Program Listing

```
 10 HOME
 20 PRINT"TO ROLL FOUR DICE, PRESS ANY
    KEY"
 30 GET K$
 40 HOME
 50 FOR L=1 TO 2
 60 GOSUB 200
 70 PRINT "**********"
 80 PRINT "* ";DL;" * ";DR;" *"
 90 PRINT "**********"
100 PRINT
110 NEXT L
120 PRINT:PRINT:GOTO 20
```

```
200 DL=INT(7*(RND(1)))
210 IF DL<1 THEN 200
220 DR=INT(7*(RND(1)))
230 IF DR<1 THEN 220
240 RETURN
```

4 Secret Message I

Secret messages can be lots of fun! They often are composed of codes in which letters of the alphabet have been replaced by numbers.

In this easy-to-use program, the computer generates a list of pseudorandom numbers and assigns one number to each letter of the alphabet. You use the numbers, in lieu of letters, to write notes to your friends.

There is very little chance of the same number being assigned to two different letters because available numbers range from zero to 999.

When typing this program into your APPLE, be sure to separate the alphabet letters with commas in line 100.

By the way, note the nice two-column screen printing format! Line 50 does that.

Program Listing

```
 10 HOME
100 DATA A,B,C,D,E,F,G,H,I,J,K,L,M,N,
    O,P,Q,R,S,T,U,V,W,X,Y,Z
200 FOR N=1 TO 13
210 C=INT(1000*(RND(1)))
220 READ L$
230 D=INT(1000*(RND(1)))
240 READ J$
250 PRINT L$;" ";C,K$;" ";D
260 NEXT N
```

5 Secret Message II

Now here's something really different in a secret message. The computer asks you for your message, removes it from the screen, stores it, and then displays it backwards!

Yes, we said SDRAWKCAB. It's like looking through your TV tube from behind.

Make up your messages as whole sentences. For instance: I WENT TO THE STORE comes out EROTS EHT OT TNEW I. To clear the old message, and put in a new one, press any key on the APPLE keyboard.

Program Listing

```
 10 HOME
 20 INPUT"WHAT IS THE MESSAGE ?  ";A$
 30 L=LEN(A$)
 40 FOR J=L+1 TO 1 STEP -1
 50 B$=B$+MID$(A$,J,1)
 60 NEXT J
 70 HOME:PRINT B$
 80 GET C$
 90 IF C$="" THEN 80
100 CLEAR:GOTO 10
```

6 Step Up

From time to time it's useful to be able to make the computer display ever-larger numbers in sequence. Here's how to step up numbers.

The subroutine in lines 500-540 generates a one-digit number. That numerical value is converted to a string and, at line 110, added to whatever already exists as K$. Thus, K$ grows by one digit at a time.

This program actually is a little game to test your ability to recall long strings of numbers. It starts with a

single digit and keeps presenting ever-longer numbers until you no longer can recall what you have just seen on the screen.

When you reach a number you cannot recall, the computer will give you only five tries. After that it ends the game and reports your score. If you wish to end the game, enter X instead of a number.

Line 130 controls the length of time of display of the number to be recalled. For more time, increase the number 500 in line 130. For less time, decrease the number 500.

At the end of the game, the computer reports the length of your last attempt.

Program Listing

```
 10 HOME:CLEAR
 20 PRINT CHR$(7)
 30 PRINT"HOW LONG A NUMBER CAN YOU
    RECALL ?"
 40 PRINT"WATCH CLOSELY..."
 50 FOR T=1 TO 1500:NEXT T
100 GOSUB 500
110 K$=K$+N$
120 PRINT:PRINT K$
130 FOR T=1 TO 500:NEXT T
140 HOME:A=A+1
150 IF A=6 THEN 400
160 PRINT"THIS IS YOUR ATTEMPT NUMBER "
    ;A
170 INPUT"WHAT WAS THAT NUMBER ?";G$
180 IF G$="X" THEN 400
190 IF G$=K$ THEN 300
200 PRINT"WRONG.  TRY AGAIN."
210 GOTO 120
300 PRINT:PRINT"YES, ";K$;" WAS THE NUMBER"
310 PRINT"NOW TRY A LONGER ONE..."
320 FOR T=1 TO 2000:NEXT T
330 A=Ø:GOTO 100
400 HOME:PRINT CHR$(7)
410 PRINT"YOU GOT OUT AS FAR AS ";D
    ;" DIGITS"
```

```
420 PRINT:PRINT"THANKS FOR TRYING"
430 END
500 N=INT(10*(RND(1)))
510 IF N<1 THEN 500
520 N$=STR$(N)
530 D=D+1
540 RETURN
```

7 Box Score

To dress up scores during and at the end of a game program, use this method of putting those scores in a box. The box around the score will highlight it and jazz up your video display.

The program here has a temporary substitute for lines 10-40. Normally, you would obtain player's name and score from some larger game program you already have on hand, or are writing. Line 20 gets from you a name and stores it in N$. Line 30 gets a score and stores it in S. If lines 50-250 were a subroutine to a larger program, you would need a RETURN at line 110.

To try the highlighting technique, type in this program just as it is here. All of it, from line 10 through line 250, and RUN it. You'll see the name and score you give the APPLE displayed in a box on the video screen.

Program Listing

```
 10 HOME
 20 INPUT "PLAYER'S NAME ? ";N$
 30 INPUT "PLAYER'S SCORE ? ";S
 40 PRINT
 50 S$=STR$(S)
 60 LN=LEN(N$)+LEN(S$)
 70 LT=LN+14
 80 GOSUB 200
 90 PRINT "* ";N$;"'S SCORE: ";S$;" *"
100 GOSUB 200
110 PRINT:PRINT:GOTO 20
```

```
120 END
200 FOR L=1 TO LT
210 AS$=AS$+"*"
220 NEXT L
230 PRINT AS$
240 A$=""
250 RETURN
```

8 Sound Off

You can make your APPLE computer beep on command!

Use the instruction PRINT CHR$(7) in a program line, such as we have done in line 40 here. The APPLE will beep!

Insert PRINT CHR$(7) in a FOR/NEXT loop to make it beep repeatedly. We've done that here to make it beep 10 times.

Program Listing

```
10 HOME
20 FOR L=1 TO 10
30 PRINT "BEEP"
40 PRINT CHR$(7)
50 NEXT L
```

9 Say Goodbye, Computer

Kids, relatives and neighbors really go bananas when your computer talks directly to them via its video screen. So, the finishing touch on your programs should be a ByeBye statement from the computer. It'll be like the cherry atop an ice cream sundae.

We have two partial programs here. In the first, the computer asks if you want to try again. If you say no, it

says it will stay alert for more enemies. Then it asks you to turn it off, to conserve electricity.

Lines 470-600 are the goodbye section of the program. By the way, line 510 is a trap to catch any answer which might be neither yes nor no. Line 560 causes the computer to ring its bell five times.

In our second program, the computer thanks you for playing. Lines 840-890 are the goodbye section.

Both of these programs, as reproduced here, are complete and can run alone. Or they would make good subroutines in a larger set of BASIC instructions. To use them as subroutines, you must adjust variable names and line numbers to match those available in your larger program.

Try typing each of these into your APPLE. They'll RUN alone and show you how to make your computer say goodbye.

Program Listing

```
 10 HOME
 20 Q=55
 30 GOTO 470
100 PRINT "MAIN PROGRAM STARTS HERE"
470 PRINT CHR$(7)
480 INPUT "DO YOU WANT TO TRY AGAIN";L$
490 IF LEFT$(L$,1)="N" THEN GOTO 520
500 IF LEFT$(L$,1)="Y" THEN PRINT
    "OKAY,  STANDBY...":GOTO 100
510 PRINT "PLEASE ANSWER YES OR NO"
    :GOTO 480
520 PRINT "OKAY"
530 PRINT "YOU BLASTED A TOTAL OF "
    ;Q;" CYLONS"
540 PRINT:PRINT "IF MORE CYLONS TURN UP,
    I WILL SOUND OFF"
550 PRINT "LIKE THIS..."
560 FOR J=1 TO 5:PRINT CHR$(7):NEXT J
570 PRINT:PRINT "NOW, PLEASE TURN ME OFF"
580 PRINT"I NEED SOME REST.  THANK YOU."
600 END
```

Program Listing

```
 10 HOME
 20 Q=55
100 PRINT:PRINT "MAIN PROGRAM STARTS HERE"
840 PRINT CHR$(7)
850 INPUT "DO YOU WANT TO PLAY AGAIN ?";X$
860 IF LEFT$(X$,1)="Y" THEN GOTO 100
870 PRINT:PRINT "OKAY. THANKS FOR PLAYING."
880 PRINT "YOUR TOTAL SCORE IS ";Q
890 END
```

10 Computer Rating Service

Of course, once you know a player's batting average it still might need some interpretation. In this program, the computer takes a look at a batting average and makes a comment.

Remember that this listing, starting here with line 800, is a partial program to be tacked on the end of a longer game. Note that, at 800, you already have values for G (number right) and E (number of tries). Line 810 converts those raw numbers to a batting average (H).

Then, the computer takes that batting average, stored in H, and compares it with values shown in lines 830 to 870. Depending upon the value of H, a slogan is selected by a jump to one of the lines 880 to 950.

By the way, check line 880. You'll see a special epitaph for players with batting averages above 900.

Program Listing

```
 10 HOME
700 G=55:E=100
800 PRINT "YOU GOT ";G;" RIGHT IN "
    ;E;" TRIES"
810 H=(G/E)*1000:INT(H):PRINT "BATTING "
    ;H
820 PRINT "YOU ARE..."
```

```
830 IF H<100 THEN GOTO 910
840 IF H<300 THEN GOTO 920
850 IF H<500 THEN GOTO 930
860 IF H<700 THEN GOTO 940
870 IF H<900 THEN GOTO 950
880 PRINT"QUALIFIED FOR THE HALL OF FAME"
    :GOTO 960
910 PRINT"VERY NEAR THE BOTTOM OF THE
    BARREL":GOTO 960
920 PRINT"POOR":GOTO 960
930 PRINT"AVERAGE":GOTO 960
940 PRINT"TOP NOTCH":GOTO 960
950 PRINT"DAMN NEAR PERFECT !"
960 PRINT"YOUR BATTING AVERAGE IS ";H
970 END
```

11 Batting Average

Once you know the number of times you were right and wrong in a game, as in Tip Number 20, it's fun to convert those raw numbers to a batting average. Numbers right and numbers wrong take on a new meaning when changed to a batting average. Folks seem to be able to understand a batting average better.

Our program, starting at line 900, is a partial listing designed to be tacked onto the end of your longer game program to display the final results of play. It will show the number of tries, number of right answers, percentage right, and batting average.

You'll want to test load this program so add lines 10 and 800 as shown. Line 800 will give you the R and T values you'll need going into the program at line 900.

Program Listing

```
 10 HOME
800 R=55:T=100
900 PRINT R;" RIGHT"
910 PRINT "IN ";T;" TRIES"
```

```
920 D=R/T:P=100*D:B=10*P
930 PRINT "THAT'S ";P;" PERCENT"
940 PRINT "BATTING ";B
```

12 Mystery Clues

Want to create your own murder mystery? Figure out whodunit and write your program backwards from there. When your players make wrong guesses, give them tantalizing clues.

Here's a short program which you can load into your computer in a matter of minutes. Key it in and try it out. It shows how you can add clues to your mysteries.

For simplicity, we assume here the Butler did it. Note that, in line 20, we are making him equal to X$. At line 30, the computer stops to ask you whom you think did it. Your answer is recorded in A$.

In line 40, your answer, lodged in A$, is compared with the computer's already-certain knowledge that the Butler did it. A$ is compared with X$. If they agree, and only if they agree, the computer displays the message, "You guessed it." If you got it right, things will end right there.

If, however, you missed it, program execution (sorry about using that word in a murder mystery!) drops to line 50 where we hear the computer, "Clue: servant." After deftly dropping that clue, the computer moves back to line 10 and runs through the whole affair another time. It will keep running through it until you answer, "Butler," in response to its question in line 30.

Program Listing

```
10 HOME
20 X$="BUTLER"
30 INPUT "WHODUNIT ?";A$
40 IF X$=A$ THEN PRINT "YOU GUESSED IT"
   :GOTO 60
50 PRINT "CLUE: SERVANT":GOTO 10
60 END
```

13 60-Second Timer

A one-minute timer can be very handy for fun-n-games. This easy-to-use clock "ticks" as it counts off seconds up to 60. When it reaches 60 seconds, it rings an alarm.

The number of seconds counted can be changed by changing the number 60 in line 20.

The clock can be calibrated by changing the number 750 in line 50. Line 50 is a time-delay loop set for approximately one second.

Lines 70-90 provide a rapid burst of five beeps when the clock reaches 60 seconds. To change the length of this alarm, change the number 70 in line 5.

Program Listing

```
10 HOME
20 FOR T=1 TO 60
30 PRINT CHR$(7)
40 PRINT T;" SECONDS"
50 FOR L=1 TO 750:NEXT L
60 NEXT T
70 FOR E=1 TO 5
80 PRINT CHR$(7)
90 NEXT E
```

14 Math Flasher

Here's the basic routine (no pun intended) for an educational flash-card program. This one is bare-bones, no frills. You can dress it up with more colorful right-n-wrong messages, opening and closing billboards, etc. You could even make it keep score and present a "batting average" at the end of its run.

Here's how it works:

Lines 10- 90 determine which type of math you wish to

do. Lines 50-80 move program action to the appropriate group of lines further along in the program.

Lines 200-280 handle addition. Lines 300-390, subtraction. Lines 400-480, multiplication. Lines 500-590, division.

For example, look at lines 200-280. Two separate random numbers are generated (lines 200 and 210). The random numbers are labeled P and Q. At line 220, the program uses P and Q and asks you to add them together. Line 230 waits for and accepts your answer.

Line 240 clears old, unwanted words from the screen to reduce confusion. At line 250, the program makes the right or wrong decision, using the powerful IF/THEN statement. Line 270 prints the correct answer.

Program execution for subtract (lines 300-390), multiply (lines 400-480), and divide (lines 500-590), are similar except for line 320 in subtraction and line 520 in division.

We make the assumption that it is not desirable to have negative numbers as results of subtraction. That is, we want only subtraction problems with results of zero, one, two, three, or higher. We want no problems which would result in answers below zero such as -1, -2, -3, and so forth. So, line 320 tests P and Q, before presenting the problem on the screen. If they will result in a negative-number answer, then the program returns to lines 300-310 for two new numbers.

In division, we want whole-number answers. That is, we want answers like 2 or 11 or 26. Not answers like 1.81 or 9.75 or 21.3343. So, line 520 tests P and Q to make sure their dividend will be a whole number. If not, the program goes back to line 500 and line 510 for two new numbers.

Program Listing

```
10 HOME:PRINT"DO YOU WANT TO"
20 PRINT"ADD":PRINT"SUBTRACT"
30 PRINT"MULTIPLY":PRINT"DIVIDE"
40 PRINT:INPUT"WHICH ?";B$
50 IF B$="ADD" THEN GOTO 200
60 IF B$="SUBTRACT" THEN GOTO 300
```

```
 70 IF B$="MULTIPLY" THEN GOTO 400
 80 IF B$="DIVIDE" THEN GOTO 500
 90 GOTO 40
200 P=INT(10*(RND(1)))
210 Q=INT(10*(RND(1)))
220 PRINT:PRINT"ADD ";P;" PLUS ";Q
230 INPUT R
240 HOME
250 IF R=P+Q THEN PRINT "CORRECT":GOTO270
260 PRINT "WRONG"
270 PRINT P;" PLUS ";Q;" EQUALS ";P+Q
280 FOR LL=1 TO 1000:NEXT LL
290 PRINT:GOTO 200
300 P=INT(10*(RND(1)))
310 Q=INT(10*(RND(1)))
320 IF P-Q<Ø THEN GOTO 300
330 PRINT:PRINT"SUBTRACT ";Q;" FROM ";P
340 INPUT R
350 HOME
360 IF R=P-Q THEN PRINT "CORRECT":GOTO 380
370 PRINT"WRONG"
380 PRINT P;" MINUS ";Q;" EQUALS ";P-Q
385 FOR LL=1 TO 1000:NEXT LL
390 PRINT:GOTO 300
400 P=INT(10*(RND(1)))
410 Q=INT(10*(RND(1)))
420 PRINT:PRINT"MULTIPLY ";P;" TIMES "
    ;Q
430 INPUT R
440 HOME
450 IF R=P*Q THEN PRINT "CORRECT"
    :GOTO 470
460 PRINT"WRONG"
470 PRINT P;" TIMES ";Q;" EQUALS ";P*Q
480 FOR LL=1 TO 1000:NEXT LL
490 PRINT:GOTO 400
500 P=INT(100*(RND(1)))
510 Q=INT(10*(RND(1)))
515 IF Q<1 THEN GOTO 510
520 IF P/Q <> INT(P/Q) THEN GOTO 500
530 PRINT "DIVIDE ";P;" BY ";Q
540 INPUT R
```

```
550 HOME
560 IF R=P/Q THEN PRINT "CORRECT"
    :GOTO 580
570 PRINT "WRONG"
580 PRINT P;" DIVIDED BY ";Q;" EQUALS "
    ;P/Q
585 FOR LL=1 TO 1000:NEXT LL
590 PRINT:GOTO 500
```

15 Sorting Scores

Here's how to sort a set of scores. Any numbers can be used. Zero is assumed to be lower than any positive number and a negative number is lower than zero.

Key in as many numbers as you like. Then key a zero when you want your APPLE to compute final results. Obviously, a zero cannot be in the set of numbers you are sorting since we use zero to get out of the input loop.

At the end of the RUN, the computer will tell you which number is lowest and which is highest.

Program Listing

```
 10 HOME
 20 INPUT"TYPE IN A SCORE";N
 30 IF N=Ø THEN 100
 40 S=S+1
 50 IF S=1 THEN LN=N:HN=N
 60 IF N<LN THEN LN=N
 70 IF N>HN THEN HN=N
 80 FOR L=1 TO 5
 90 SOUND=PEEK(-16336)
100 NEXT L
110 GOTO 10
120 HOME
130 PRINT"LOW NUMBER IS ";LN
140 PRINT
150 PRINT"HIGH NUMBER IS ";HN
160 END
```

Sample Run

```
        RUN       RETURN

        TYPE IN A SCORE
BEEP    22        RETURN
BEEP    55        RETURN
BEEP    77        RETURN
BEEP    11        RETURN
         Ø        RETURN

        LOW NUMBER IS 11
        HIGH NUMBER IS 77
```

16 Name In A Box

Put your name up in lights! Or, at least, on the video display screen of your APPLE computer.

This short program creates a box on the screen and puts a name you have specified into that box. The name is highlighted.

You can change what the box is composed of by changing the asterisks in lines 60, 100 and 120.

Program Listing

```
 10 HOME
 20 INPUT"WHAT IS YOUR FIRST NAME ? ";N$
 30 LN=LEN(N$)
 40 LT=LN+4
 50 FOR L=1 TO LT
 60 AS$=AS$+"*"
 70 NEXT L
 80 PRINT AS$
 90 AS$=""
100 PRINT "* ";N$;" *"
110 FOR L=1 TO LT
120 AS$=AS$+"*"
```

```
130 NEXT L
140 PRINT AS$
150 AS$=""
160 PRINT:PRINT:GOTO 20
```

17 Audible Timer I

Here's a different sound effect for your game programs. This brief set of program lines causes your APPLE to present a continuous audible buzz with a sharp beep every second.

You can set the time faster or slower by changing the number 100 in line 50. To make the clock run slower, increase the number 100 or to speed up the clock, decrease the number 100 in line 50.

Program Listing

```
 10 HOME:PRINT CHR$(7)
 20 PRINT TAB(10) "AUDIBLE TIMER"
 30 PRINT:INPUT"HOW MANY SECONDS DO
    YOU WANT ";S
 40 FOR T=1 TO S
 50 FOR L=1 TO 100
 60 BZ=PEEK(-16336)
 70 NEXT L
 80 PRINT CHR$(7)
 90 NEXT T
100 PRINT"TIME IS UP"
110 PRINT:PRINT"TO GO AGAIN, PRESS ANY
    KEY"
120 GET KY$
130 GOTO 10
```

18 Audible Timer II

This is a variation on the previous timer program. This set of instructions causes your APPLE computer to make an interesting rapid continuous clock-style tick with a beep every second.

Adjust the beep time by increasing or decreasing the number eight in line 20.

Program Listing

```
10 HOME
20 FOR T=1 TO 8
30 FOR L=1 TO 2
40 BZ=PEEK(-16336)
50 NEXT L
60 FOR L=1 TO 100:NEXT L
70 NEXT T
80 PRINT CHR$(7)
90 GOTO 20
```

19 Traditional Dice Roll

Here's a simple, brief way to roll and display results for two dice.

Lines 100-110 get a random number between 1 and 6 and store it in A. Lines 200-210 get another random number from 1 to 6 and store it in B.

Lines 300-310 print the contents of A and B along with a suitable message. Lines 400-430 cause the computer to await the press of any key on its keyboard. When a key is pressed, dice are rolled and new results presented on the video display.

Program Listing

```
 10 HOME
100 A=INT(10*(RND(1)))
```

```
110 IF A<1 OR A>6 THEN GOTO 100
200 B=INT(10*(RND(1)))
210 IF B<1 OR B>6 THEN GOTO 200
300 PRINT "FIRST DICE:  ";A
310 PRINT "SECOND DICE: ";B
400 X=PEEK(-16384)
410 POKE -16368,Ø
420 IF X<128 THEN GOTO 400
430 PRINT:GOTO 100
```

20 Keeping Game Scores

Writing a computer football game? Spelling bee? Cave adventure? No matter what kind of fun you are preparing, you'll need a way to keep score. Here's how.

The wealthy English duke has just been killed in our little mystery game. In lines 10 through 160 of our program listing, below, you play the game, attempting to find out whodunit.

The trick here is in the scorekeeping. Note line 170. If you guessed correctly in response to the query in line 160, at line 170 the computer will give you credit by adding one point to your score stored in memory location R. It does that by comparing your line 160 answer stored in P$ with the correct answer stored in A$.

If you blew it and guessed wrong, the program drops below line 170 to line 180 where it increases your "wrong score" by adding one point to W.

If you got a W+1 at line 180, the program moves back to line 100 and gets you to try again. If you scored a victory and got an R+1 at line 170, the program jumps to line 200 where it stops to display your total right and wrong score. After that, it's back to line 20 for a complete new run-through.

Program Listing

```
 10 HOME
 20 PRINT CHR$(7)
 30 S=INT(10*(RND(1)))
 40 IF S>6 THEN GOTO 30
 50 IF S<1 THEN GOTO 30
 60 FOR L=1 TO S
 70 READ A$
 80 NEXT L
 90 RESTORE
100 PRINT"WHO KILLED THE DUKE ?"
110 READ B$,C$,D$,E$,F$,G$
120 RESTORE
130 PRINT"WAS IT THE..."
140 PRINT B$,C$,D$,E$,F$,G$
150 PRINT CHR$(7)
160 INPUT "WHODUNIT ?";P$
170 IF A$=P$ THEN R=R+1:PRINT CHR$(7)
    :PRINT CHR$(7):PRINT"RIGHT !":GOTO 200
180 PRINT"NO !  NOT THE ";P$:W=W+1
    :PRINT:GOTO 100
200 PRINT
210 PRINT "YOUR SCORE IS..."
220 PRINT R;" RIGHT AND ";W;" WRONG"
230 PRINT:GOTO 20
300 DATA BUTLER,NANNY,GARDNER,BURGLAR,SON,
    WIFE
```

21 Winner's Message

Flashing lights. Ringing bells. This message to hail a win in a game really attracts attention!

Use it at an appropriate point in your game. It will print the message YOU WIN on the video screen with a solid-bar underline, and ring a bell. The message flashes. The underline does not.

Here's how it works: lines 30 to 80 print the message.

Lines 90 to 110 display the underline bar. Line 120 rings the bell.

The number 25 in line 20 controls the number of flashes and bell rings. You may change that number if you like.

You may change the message, if you like. Use the letter and number table you see here to select a new message. The message line is 40 spaces in memory locations 1192 to 1231. The underline is in locations 1320 to 1359. Lengthen the underline bar by changing the range in line 90, up to a maximum of L = 1320 TO 1359.

A	65	T	84
B	66	U	85
C	67	V	86
D	68	W	87
E	69	X	88
F	70	Y	89
G	71	Z	90
H	72		
I	73		
J	74	Ø	48
K	75	1	49
L	76	2	50
M	77	3	51
N	78	4	52
O	79	5	53
P	80	6	54
Q	81	7	55
R	82	8	56
S	83	9	57

Change the number of bell rings and number of flashes by changing the number 25 in line 20. This program will make a good subroutine in a larger set of instructions.

Program Listing

```
10 HOME
20 FOR N=1 TO 25
30 POKE 1208,89
```

```
 40 POKE 1209,79
 50 POKE 1210,85
 60 POKE 1212,87
 70 POKE 1213,73
 80 POKE 1214,78
 90 FOR L=1335 TO 1343
100 POKE L,47
110 NEXT L
120 PRINT CHR$(7)
130 NEXT N
140 HOME
```

22 Ring The Bell !

Here's another, slightly different version of our popular APPLE bell ringer program. This one allows you to set a time delay between rings.

We use the handy PRINT CHR$(7) instruction to do the actual ringing. The N value in line 20 sets the actual number of repeat rings. (The rings might better be described as beeps.)

The L value in line 40 establishes the time delay between rings. To insert more time, increase the number 100 in line 40. To shorten the time between beeps, lower the number 100 in line 40.

Program Listing

```
10 HOME
20 FOR N=1 TO 5
30 PRINT CHR$(7)
40 FOR L=1 TO 100:NEXT L
50 NEXT N
```

23 Question & Answer

Here's how to use the DATA statement, and the computer's ability to search for data, to create a Q&A.

We put DATA in lines 20-130. It could be anywhere in the program. For instance, at the end at lines 400-510.

The computer sees two items in each data line. Program lines 140 and 160 force the machine to take only odd-numbered data from the list. That is, S$ in line 180 is always the first piece of data in a data line. And C$ in line 210 is always the second item in a data line. Line 230 checks to see if you answered the line 220 question correctly.

Program Listing

```
 10 HOME
 20 DATA JANUARY,31
 30 DATA FEBRUARY,28
 40 DATA MARCH,31
 50 DATA APRIL,30
 60 DATA MAY,31
 70 DATA JUNE,30
 80 DATA JULY,31
 90 DATA AUGUST,31
100 DATA SEPTEMBER,30
110 DATA OCTOBER,31
120 DATA NOVEMBER,30
130 DATA DECEMBER,31
140 R=INT(100*(RND(1)))
150 IF R>24 THEN GOTO 140
160 IF INT(R/2)=R/2 THEN R=R-1
170 FOR L=1 TO R
180 READ S$
190 NEXT L
200 PRINT "MONTH IS ";S$
210 READ C$
220 INPUT "HOW MANY DAYS ";D$
230 IF D$=C$ THEN PRINT "CORRECT"
    :GOTO 300
```

```
240 PRINT "WRONG"
300 PRINT "NUMBER OF DAYS IS ";C$
310 RESTORE
320 PRINT
330 GOTO 140
```

24 Timer

Good for home self-testing as well as in the classroom. Place your computer in the corner and let it time your exams.

After you press any key to start it, it prints a continuously updated display of time, including minutes and seconds.

Program Listing

```
 10 HOME:SP=16
 20 PRINT"EVENT TIMER":PRINT
 30 PRINT"HOW MANY MINUTES"
 40 INPUT"TO THE END OF THE EVENT ";LT
 50 PRINT:PRINT"PRESS ANY KEY"
 60 PRINT"TO START TIMING"
 70 GET ST$
100 HOME
110 C=C+1
120 IF C>(SP*LT*60) THEN 300
130 MN=INT(C/SP/60)
140 SC=INT((C/SP)-(60*MN))
150 PRINT MN;" MINUTES",SC;" SECONDS"
160 GOTO 110
300 HOME:PRINT CHR$(7)
310 PRINT"TIME IS UP"
320 PRINT LT;" MINUTES HAVE PASSED"
330 PRINT:PRINT"TO TIME AGAIN, PRESS
    ANY KEY"
340 CLEAR:GET TA$
350 GOTO 10
```

25 The Original Hi-Lo Game

Here it is. Where everybody started in microcomputer programming back in the Seventies. The first game ever played was a high-low guess-the-number routine.

The computer selects a secret number. You try to guess it. The computer tells you whether or not you are too high, too low, or right on the number.

Here's how it works: the secret number can be zero to 1000. Line 100 generates a random number (the secret number) and stores it. Line 210 asks you to guess the number.

Lines 300-310 decide if you are right or wrong. Line 220 keeps track of the number of attempts.

Program Listing

```
 10 HOME:T=Ø
100 R=INT(1000*(RND(1)))
200 PRINT CHR$(7)
210 INPUT"GUESS THE NUMBER";B
220 T=T+1:PRINT"THAT WAS TRY NUMBER ";T
300 IF B>R THEN PRINT CHR$(7)
    :INPUT"TOO HIGH. GUESS AGAIN.";B
    :GOTO 220
310 IF B<R THEN PRINT CHR$(7)
    :INPUT"TOO LOW. GUESS AGAIN.";B
    :GOTO 220
400 PRINT CHR$(7):PRINT CHR$(7):PRINT"YES !"
410 PRINT R;" IS THE NUMBER"
420 PRINT"YOU GOT IT IN ";T;" TRIES"
500 PRINT:T=Ø:GOTO 100
```

Text on Text

26 Killing Time

Sometimes, it may seem to you as if the computer will never get to the result of a job. You understand the processing delay but your non-computer friends may not. They could be confused by the wait and think the computer is "broken."

To keep their minds off the slowness, give them something to look at while the computer is "thinking."

The added, extra lines, numbered 50, 60, 70 and 80, take up more processing time but make for less confusion. Computing may take a bit longer but your fun will be increased.

If you delete lines 50-80 you'll see how the program runs faster but the blank screen is confusing.

Program Listing

```
 10 HOME
 20 INPUT"GIVE ME A NUMBER";N
 30 FOR L=1 TO N
 40 X=X+L
 50 FOR T=1 TO 100:NEXT T
 60 HOME
 70 FOR T=1 TO 100:NEXT T
 80 PRINT "I AM THINKING"
 90 NEXT L
100 HOME:PRINT CHR$(7)
110 PRINT"I HAVE THE ANSWER"
120 PRINT"THE TOTAL OF ALL NUMBERS"
130 PRINT"FROM 1 TO ";N;" IS ";X
200 PRINT:CLEAR:GOTO 20
```

27 Number-Error Trapping

Good programs, those which are well written, need *error trapping*. It's a technique for making sure persons communicating with the computer don't key in inap-

propriate data or make mistakes which would cause computation problems for the computer.

For instance, see the example program here. In line 10 the computer asks for a number. In line 20, if the number is too low, it says so and goes back to line 10 to repeat its request.

At line 30, if the number received at line 10 is too large, it says so and goes back to line 10 for a better choice.

The result is only printed at line 40 when a satisfactory number has been keyed in back at line 10.

You can set your own limits by changing the 10 in line 20 and the 100 in line 30.

Program Listing

```
 5 HOME
10 INPUT"GIVE ME A NUMBER";A
20 IF A<10 THEN PRINT "TOO LOW":GOTO 10
30 IF A>100 THEN PRINT "TOO HIGH"
   :GOTO 10
40 PRINT A
```

28 Word-Error Trapping

The same kind of error trapping is available for strings. Suppose the program, as in this example, asks at line 10 for a word. It is looking for YES or NO. If it gets a YES, then line 20 sees that it got what it wanted and moves operations along to line 100.

If it gets a NO, then line 20 hasn't received what it wants so program execution moves on to line 30. Here, at line 30, the program finds something useful and shoots operations down to line 200.

If, however, neither YES nor NO were entered at line 10, then neither lines 20 nor 30 would be satisfied so action would drop to line 40. Here, the error is trapped by commanding the operator to give one of the two correct answers. Then, at line 50, the operation is returned to line 10 for a new try at the correct input.

Program Listing

```
  5 HOME
 10 INPUT"WANT TO PLAY AGAIN ?";A$
 20 IF A$="YES" THEN 100
 30 IF A$="NO" THEN 200
 40 PRINT"PLEASE ANSWER ONLY YES OR NO"
 50 GOTO 10
100 PRINT A$
110 END
200 PRINT A$
210 END
```

29 Y or N

Here's a handy use for the powerful LEFT$ instruction. Make the computer decide whether you have answered yes or no to a question.

The tests in line 20 and 30 look at the first letter of your answer to the question in line 10. If the first letter is Y then line 20 shoots program action on to line 100. If the first letter of your answer is N, then line 20 does nothing and action drops to line 30 where the program jumps to line 300.

If the first letter of your answer was neither Y nor N, then the program passes lines 20 and 30 and, at line 40, tells you that you must answer yes or no. Then line 50 pushes action back to line 10 where the entire process starts over.

You only can answer with words starting with Y or N. Another word will be "trapped" out.

Program Listing

```
  5 HOME
 10 INPUT"WANT TO PLAY AGAIN ?";A$
 20 IF LEFT$(A$,1)="Y"THEN 100
 30 IF LEFT$(A$,1)="N"THEN 200
 40 PRINT"YOU MAY ANSWER Y OR N OR
    YES OR NO"
```

```
 50 GOTO 10
100 PRINT "YES"
110 END
200 PRINT "NO"
210 END
```

30 Categorizing

A large quantity of numbers can be categorized and thereby cut down into a smaller quantity of numbers. See our example: it takes test scores and divides them into ranges labeled A, B, C, D, and F.

The program assumes exam or test scores in a range of zero to 100. The letter grades include zero to 59, F; 60-69, D; 71-79, C; 80-89, B; 90-100, A.

Key in as many scores as you like and then enter the letter X to stop the entry cycle.

Lines 100-140 sort all scores into the A through F categories. Lines 150-170 sort highest and lowest scores. Line 200 finds mid-range and average scores.

Program Listing

```
 10 HOME:CLEAR
 20 PRINT CHR$(7)
 30 PRINT "ENTER A GROUP OF SCORES"
 40 PRINT "FROM ZERO TO 100, ONE AT A
    TIME"
 50 PRINT "ENTER X AFTER LAST SCORE"
 60 PRINT: INPUT"SCORE = ";G$
 70 IF G$="X" THEN 200
 80 G=VAL(G$)
 90 N=N+1
100 IF G<60 THEN F=F+1:GOTO 150
110 IF G<70 THEN D=D+1:GOTO 150
120 IF G<80 THEN C=C+1:GOTO 150
130 IF G<90 THEN B=B+1:GOTO 150
140 A=A+1
150 IF N=1 THEN L=G:H=G
```

```
160 IF G<L THEN L=G
170 IF G>H THEN H=G
180 S=S+G
190 GOTO 60
200 P=S/N:M=L+((H-L)/2)
210 HOME:PRINT"THERE WAS A TOTAL OF"
    ;N;" SCORES"
220 PRINT"RANGING FROM ";L;" TO ";H
230 PRINT"MID-RANGE SCORE WAS ";M
240 PRINT"AVERAGE SCORE WAS ";P
250 PRINT:PRINT"TOTALS FOR EACH LETTER
    GRADE:"
260 PRINT"A: ";A
270 PRINT"B: ";B
280 PRINT"C: ";C
290 PRINT"D: ";D
300 PRINT"F: ";F
310 PRINT:CLEAR:GOTO 20
```

31 Look-Up Table

You can use the computer as a standard look-up reference. This program, for photographers, gives filter factors for exposure corrections.

The appropriate corrections are stored ahead of time in lines 250 to 360 and 430 to 540 for use later in the field.

Serious photographers, amateur and pro, carry an assortment of lens filters in their bags. Lens filters absorb light so cameramen must make corrections when using them.

The *filter factor* is used in making that correction. The photographer sets the exposure meter for the speed of the film in use without a filter. Then, he modifies the camera settings indicated by the meter. For instance, suppose the factor is 2 and the setting indicated by the meter is 1/125 second at f/11. The photographer opens the lens by one full f/stop to f/8.

This computer program displays the proper f/stop

correction when factor is known or factor when correction is known.

Program Listing

```
 10 HOME
 20 PRINT"TO COMPUTE A NEW FILTER FACTOR"
 30 PRINT"FOLLOWING AN F/STOP CHANGE"
 40 PRINT"PRESS 1 AND RETURN"
 50 PRINT:PRINT"TO FIND HOW MUCH TO
    CHANGE THE F/STOP"
 60 PRINT"AFTER CHANGING FILTERS"
 70 PRINT"PRESS 2 AND RETURN"
 80 INPUT Z
 90 IF Z<1 OR Z>2 THEN GOTO 80
100 ON Z GOTO 200,400
200 HOME:PRINT CHR$(7)
210 W=Ø
220 PRINT"YOU NEED TO COMPUTE A"
230 PRINT"NEW FILTER FACTOR BECAUSE"
240 INPUT"YOU HAVE CHANGED THE F/STOP
    BY +";Y$
250 IF Y$="1/3" THEN W=1.2
260 IF Y$="2/3" THEN W=1.5
270 IF Y$="1" THEN W=2
280 IF Y$="1 1/3" THEN W=2.5
290 IF Y$="1 2/3" THEN W=3
300 IF Y$="2" THEN W=4
310 IF Y$="2 1/3" THEN W=5
320 IF Y$="2 2/3" THEN W=6
330 IF Y$="3" THEN W=8
340 IF Y$="3 1/3" THEN W=10
350 IF Y$="3 2/3" THEN W=12
360 IF Y$="4" THEN W=16
370 PRINT"FILTER FACTOR SHOULD BE ";W
380 PRINT:GOTO 210
400 HOME:PRINT CHR$(7)
410 V$="?"
420 INPUT"THE FILTER FACTOR IS ";X
430 IF X=1.2 THEN V$="1/3"
440 IF X=1.5 THEN V$="2/3"
450 IF X=2 THEN V$="1"
```

```
460 IF X=2.5 THEN V$="1 1/3"
470 IF X=3 THEN V$="1 2/3"
480 IF X=4 THEN V$="2"
490 IF X=5 THEN V$="2 1/3"
500 IF X=6 THEN V$="2 2/3"
510 IF X=8 THEN V$="3"
520 IF X=10 THEN V$="3 1/3"
530 IF X=12 THEN V$="3 2/3"
540 IF X=16 THEN V$="4"
550 PRINT "YOU CHANGE THE F/STOP BY +"
    ;V$
560 PRINT:GOTO 410
```

32 Entering: Letter Stop

One way to conclude an input series, and get out of its entry loop, is to use a key letter to promote a jump. In this brief example, we input numbers, at line 100, as string values. If we give the computer an X rather than a number, it will jump down to line 200 for new action.

Numbers keyed in are stored first as strings. Then line 120 changes them to number values for the addition in line 130.

Program Listing

```
 10 HOME
 20 B=Ø:C=Ø
100 INPUT"GIVE ME A NUMBER";A$
110 IF A$="X" THEN GOTO 200
120 B=VAL(A$)
130 C=C+B
140 GOTO 100
200 PRINT:PRINT CHR$(7)
210 PRINT "THE TOTAL OF ALL THOSE
    NUMBERS IS ";C
```

33 Entering: Zero Stop

Here's another way to conclude an entry loop: have the computer be on the lookout for a plain zero. When a zero is entered, the computer will jump out of the entry cycle and on to further action.

This program totals numbers as they are added and accumulates them in memory location B. If one of the numbers entered is a zero alone, then line 110 will spot it and send the computer on down to line 200, breaking the entry cycle.

Naturally, you can't use a zero in a string of numbers to be added since zero causes the computer to quit entering and get on with displaying.

Program Listing

```
 10 HOME
 20 B=Ø
100 INPUT "GIVE ME A NUMBER PLEASE";A
110 IF A=Ø THEN GOTO 200
120 B=B+A
130 GOTO 100
200 PRINT:PRINT CHR$(7)
210 PRINT"THE TOTAL OF ALL THOSE NUMBERS
    IS ";B
```

34 Keyboard Scanner

This program demonstrates the APPLE's ability to make a one-key selection *without* a need for you to press the RETURN key. It uses the powerful GET instruction.

At line 100, the GET instruction forces the computer to await a key press and then store the value of that key in KY$. For instance, if you pressed the key for the letter A, the keyboard scanner would see that and store the letter A in memory location KY$. If you pressed X, it would store X in KY$.

The computer, after a key is pressed, examines the contents of KY$. At line 110 we find the computer looking

for an A in KY$. At line 120 we find it looking for a B there. At 130 it is looking for a C. If none are found, line 140 pushes action back up to line 100 where the computer patiently awaits another key press. Line 140, then, is an error trap to make sure no letter, other than A, B, or C, gets through.

Program Listing

```
 10 HOME
 20 PRINT"TO GO TO:"
 30 PRINT:PRINT"LINE 200","PRESS A"
 40 PRINT"LINE 300","PRESS B"
 50 PRINT"LINE 400","PRESS C"
 60 PRINT:PRINT
100 GET KY$
110 IF KY$="A" THEN 200
120 IF KY$="B" THEN 300
130 IF KY$="C" THEN 400
140 GOTO 100
200 PRINT"AT LINE 200"
210 GOTO 60
300 PRINT"AT LINE 300"
310 GOTO 60
400 PRINT"AT LINE 400"
410 GOTO 60
```

35 Character-ASCII Review

Here's a fast conversion, either to or from an ASCII value.

Lines 20-70 let you select which direction your conversion will go. If you have an ASCII value, program execution will move to line 100. The conversion is made in line 140. If you wish to convert a keyboard character to its ASCII value, the program jumps from line 60 to line 200. The conversion is done at line 230.

Program Listing

```
 10 HOME
 20 PRINT "SELECT: ASCII-TO-CHARACTER
    (PRESS A)"
 30 PRINT "OR SELECT: CHARACTER-TO-ASCII
    (PRESS C)"
 40 INPUT "WHICH ?";C$
 50 IF C$="A" THEN 100
 60 IF C$="C" THEN 200
 70 GOTO 40
100 INPUT "WHAT IS THE ASCII VALUE ?"
    ;A
110 IF A>255 THEN 100
120 HOME
130 PRINT "THE ASCII NUMBER IS ";A
140 PRINT "THE CHARACTER IS ";CHR$(A)
150 PRINT
160 GOTO 20
200 INPUT "WHAT IS THE CHARACTER ?"
    ;CH$
210 HOME
220 PRINT "THE CHARACTER IS ";CH$
230 PRINT "THE ASCII NUMBER IS "
    ;ASC(CH$)
240 PRINT
250 GOTO 20
```

36 Character Numbers

This brief program displays the ASCII value for each keyboard character, side-by-side with the character it stands for. You will be able quickly to tell what each number prints.

Line 40 is a timing loop to slow down the presentation so you can digest the information. To make it even slower, increase the number 400 in line 40. To make it faster, decrease the number 400 in line 40.

The computer's chime will ring after it reaches 255. Line 60 provides the beep. By the way, it also will beep when N = 70 in line 30 and when N = 135 in line 30.

Program Listing

```
10 HOME
20 FOR N=Ø TO 255
30 PRINT N,CHR$(N)
40 FOR L=1 TO 400:NEXT L
50 NEXT N
60 PRINT CHR$(7)
```

37 Memory Peek

Want a quick look into a specific memory location? Use the interesting PEEK instruction.

This short program allows a search of memory locations. You should change the numbers in line 20 to encompass whatever memory locations you wish to search. As written, the program will cause the computer to reveal the contents of locations 1000 to 1100. It could just as well be 100 to 200 for which you would substitute the expression:

FOR M = 100 TO 200

or 10,000 to 10,100 for which you would use the expression:

FOR M = 10000 TO 10100

These substitutions are at line 20, of course. In fact, you could look at quite a lot of locations like this:

FOR M = Ø TO 60000

Program Listing

```
10 HOME
20 FOR M=1000 TO 1100
30 P=PEEK (M)
40 C$=CHR$ (P)
50 PRINT "ASCII VALUE AT ";M;" IS "
   ;P
60 PRINT "CHARACTER AT ";M;" IS ";C$
70 PRINT
80 FOR L=1 TO 500:NEXT L
90 NEXT M
```

38 Search & Order

This program shows a way to sort information into a specific order. We are putting months, in this example, into chronological order.

You key in any quantity of months, from one to twelve, and the computer will sort and print a list. Be sure to spell each month's name fully and correctly.

This is a good demonstration of the LEFT$ function. We use LEFT$(M$,3) to get at the first three letters in the name of each month. We have to use three letters, by the way, since fewer would provide duplicates and the sort would come out wrong.

Program Listing

```
 10 HOME
 20 DIM M$(12)
 30 FOR L=1 TO 12
 40 INPUT "MONTH: ";MN$
 50 INPUT "DAYS IN THAT MONTH: ";DY$
 55 M$(L)=MN$+" HAS "+DY$+" DAYS"
 60 NEXT L
 70 FOR L=1 TO 12
 80 IF LEFT$(M$(L),3)="JAN" THEN PRINT
    M$(L)
 90 NEXT L
100 FOR L=1 TO 12
110 IF LEFT$(M$(L),3)="FEB" THEN PRINT
    M$(L)
120 NEXT L
130 FOR L=1 TO 12
140 IF LEFT$(M$(L),3)="MAR" THEN PRINT
    M$(L)
150 NEXT L
160 FOR L=1 TO 12
170 IF LEFT$(M$(L),3)="APR" THEN PRINT
    M$(L)
180 NEXT L
```

```
190 FOR L=1 TO 12
200 IF LEFT$(M$(L),3)="MAY" THEN PRINT
    M$(L)
210 NEXT L
220 FOR L=1 TO 12
230 IF LEFT$(M$(L),3)="JUN" THEN PRINT
    M$(L)
240 NEXT L
250 FOR L=1 TO 12
260 IF LEFT$(M$(L),3)="JUL" THEN PRINT
    M$(L)
270 NEXT L
280 FOR L=1 TO 12
290 IF LEFT$(M$(L),3)="AUG" THEN PRINT
    M$(L)
300 NEXT L
310 FOR L=1 TO 12
320 IF LEFT$(M$(L),3)="SEP" THEN PRINT
    M$(L)
330 NEXT L
340 FOR L=1 TO 12
350 IF LEFT$(M$(L),3)="OCT" THEN PRINT
    M$(L)
360 NEXT L
370 FOR L=1 TO 12
380 IF LEFT$(M$(L),3)="NOV" THEN PRINT
    M$(L)
390 NEXT L
400 FOR L=1 TO 12
410 IF LEFT$(M$(L),3="DEC" THEN PRINT
    M$(L)
420 NEXT L
```

39 Bubble Sort

Here's a handy routine, accepting lists of up to 100 names and sorting them into alphabetical order. The size of the list is controlled by the DIM instruction in line 20. You can change it.

Note that the screen goes blank while sorting is going on inside your APPLE so you must be patient. The more names to sort, the longer it takes.

Of course, you can change names to other categories such as items, products, units, etc.

You will have to enter 100 names as the program is written. If you wish to enter a smaller quantity of names at one time, add this line 45 to the program:

45 IF M$(L)="X" THEN GOTO 60

Line 45 uses the letter X, when typed in response to the NAME query, to force an exit from the entry cycle. Problem: the X will be sorted with your other entries and will appear in your final list! Can you think of a way to get rid of that little problem?

The program sorts by whatever is first in the input line.

Program Listing

```
 10 HOME
 20 DIM M$(100)
 30 FOR L=1 TO 100
 40 INPUT "NAME: ";M$(L)
 50 NEXT L
 60 HOME
 70 T=Ø
 80 FOR L=1 TO 99
 90 IF M$(L)<=M$(L+1) THEN 110
100 E$=M$(L):M$(L)=M$(L+1):M$(L+1)=E$
    :T=1
110 NEXT L
120 IF T=1 THEN 70
130 FOR L=1 TO 100
140 IF M$(L)<>"" THEN PRINT M$(L)
150 NEXT L
```

40 Simple Screen Clear

There is nothing more unsightly than a video screen full of miscellaneous garbage. It can be very distracting

when you start to RUN a new program while unwanted old bits and pieces of information dot the screen.

Now you can make any program clear away the unwanted junk *before* showing you results of a new RUN. Use the instruction HOME to clear the video display.

Use the HOME command to clear the screen at the start of each program. Try our sample program. Before running it, key in the following:

GR RETURN
TEXT RETURN

Now the video screen will be filled with junk ampersands. Key in RUN and see the program clear away those unwanted symbols. The alphabet is printed on a clean slate as a result of this program operation.

Program Listing

```
10 HOME
20 PRINT"ABCDEFGHIJKLMNOPQRSTUVWXYZ"
30 END
```

41 News Ticker

It's Times Square all over again! This program simulates the crawling message of a news wire.

First the computer will ask for your message. Then it will display that message endlessly (until you press the RESET key).

To speed up, or slow down, the crawl, increase or decrease the number 200 in line 90.

For a test, in answer to the computer's query, "What is the news?", try typing in: The mayor of Smithville today announced he will retire next December.

Program Listing

```
10 HOME
20 PRINT"WHAT IS THE NEWS ?"
30 INPUT N$
```

```
 40 HOME
 50 L=LEN(N$)
 60 FOR Y=1 TO L
 70 B$=B$+MID$(N$,Y,1)
 80 PRINT B$
 90 FOR T=1 TO 200:NEXT T
100 HOME
110 NEXT Y
120 B$="":GOTO 80
```

42 Create A Quiz

One of the most fascinating uses for your APPLE is in having it carry on a video conversation with your friends, relatives and neighbors. One useful way to promote such conversation is through a quiz. An instructional, educational quiz, such as we have here.

Quiz data—the computer's storehouse of knowledge—is in lines 20 to 70. Be careful, when you type them into your APPLE, to include the commas separating the two halfs of each data line. Spelling and spacing must be exact.

Lines 90 and 100 obtain a random number in the range of 1 to 11. Line 110 selects the data line for a question. Lines 120 through 140 get the appropriate word FIRST, SECOND, THIRD, FOURTH, FIFTH or SIXTH from the selected data line. Lines 160 to 180 print the quiz question on the screen, while line 190 reads the DATA line to learn the correct answer. You provide your response when the computer asks for it at line 200. Lines 220-240 decide whether you are right or wrong.

Of course, the quiz can be made much longer. In this example, it could be expanded to encompass all past U.S. presidents.

Program Listing

```
10 HOME
20 DATA FIRST,GEORGE WASHINGTON
30 DATA SECOND,JOHN ADAMS
```

```
 40 DATA THIRD,THOMAS JEFFERSON
 50 DATA FOURTH, JAMES MADISON
 60 DATA FIFTH,JAMES MONROE
 70 DATA SIXTH,JOHN QUINCY ADAMS
 80 PRINT"HOW MANY U.S. PRESIDENTS
    CAN YOU NAME"
 90 R=INT(12*(RND(1)))
100 IF R<1 THEN 90
110 IF INT(R/2)=R/2 THEN R=R+1
120 FOR L=1 TO R
130 READ S$
140 NEXT L
150 PRINT:PRINT
160 PRINT"WHO WAS THE "
170 PRINT S$
180 PRINT"PRESIDENT OF THE UNITED STATES"
190 READ C$
200 INPUT D$
210 PRINT
220 IF D$=C$ THEN PRINT "CORRECT"
    :GOTO 240
230 PRINT "WRONG"
240 PRINT "THE ";S$;" PRESIDENT WAS ";C$
250 RESTORE
260 PRINT:PRINT
270 GOTO 90
```

Gee Whiz

43 Gee Whiz I: Smart Adder

These six programs, in this section of the book, make up our *Gee Whiz* series. One of the fun ways to use your APPLE is in wowing your friends. Next time they ask, "But, what can it do?", show them its uncanny abilities at adding, spelling, writing upside down, even cracking jokes. Try these six *Gee Whiz* programs on your friends. You'll love their reactions.

Smart Adder is the first in the series. When your neighbor drops in for a cup of coffee, bring out the APPLE for a demonstration of its lightning speed.

This program adds long strings of numbers in a flash. You give the computer a number. It starts at 1 and adds all numbers up to and including your number. For instance, if you give it a five, it will add 1 plus 2 plus 3 plus 4 plus 5 and display the result.

Ask your neighbor how fast he or she can add all the numbers to 100. It should take several minutes. While he's working on it, let your APPLE do it in a split second. Your neighbor's reaction is bound to be, "Gee whiz!"

Program Listing

```
10 HOME
20 INPUT"GIVE ME A NUMBER";N
30 FOR L=1 TO N:X=X+L:NEXT L
40 PRINT CHR$(7)
50 PRINT "THE TOTAL OF 1 TO ";N
   ;" IS ";X
60 PRINT:CLEAR:GOTO 20
```

44 Gee Whiz II: Alphabet Spotter

There are 26 letters in the alphabet. Each has a number. For instance, number 1 is A. Number 20 is T. This *Gee Whiz* program has the computer ask you for a number

from 1 to 26 and then, faster than a jackrabbit, tell you what letter it goes with.

Naturally, you'll know how it works but to your non-computer friends it will seem like the APPLE is a genius!

Program Listing

```
 10 HOME
 20 DATA A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,
    P,Q,R,S,T,U,V,W,X,Y,Z
 30 PRINT"GIVE ME THE NUMBER OF"
 40 PRINT"A LETTER FROM THE ALPHABET"
 50 INPUT"FROM 1 TO 26  ";N
 60 FOR L=1 TO N
 70 READ A$
 80 NEXT L
 90 PRINT:PRINT:PRINT CHR$(7)
100 PRINT"LETTER NUMBER ";N;" IS ";A$
110 RESTORE:PRINT:GOTO 20
```

45 Gee Whiz III: Up, Down, Back, Forth

"Good golly, what can't that APPLE do?", will be the question from your surprised neighbor when you show him this neat trick.

You type in any word. The computer instantly prints it on the video display, both up and down vertically, and backward and forward horizontally. It's great to show how smart your computer is when it comes to spelling!

Program Listing

```
10 HOME
20 DIM X$(100)
30 INPUT"GIVE ME A WORD  ";A$
40 L=LEN(A$)
50 FOR J=1 TO L+1
60 X$(J)=MID$(A$,J,1)
```

```
 70 NEXT J
 80 PRINT:PRINT "DOWN:","UP:"
 90 FOR J=1 TO L+1
100 PRINT X$(J),X$(L+1-J)
110 NEXT J
120 PRINT "FORWARD:"
130 PRINT A$
140 FOR J=L+1 TO 1 STEP -1
150 Z$=Z$+MID$(A$,J,1)
160 NEXT J
170 PRINT:PRINT "BACKWARD:"
180 PRINT Z$
190 PRINT:PRINT:CLEAR:GOTO 20
```

46 Gee Whiz IV: Three-Digit Mystery

Have your neighbor secretly select any three-digit number in which all three digits are the same. Then have him tell the computer only the *sum* of those three digits. The computer will identify his secret number!

Program Listing

```
10 HOME:CLEAR
20 PRINT"SELECT A THREE-DIGIT NUMBER"
30 PRINT"WITH ALL THREE DIGITS THE SAME"
40 PRINT:PRINT"ADD THE THREE DIGITS
   TOGETHER"
50 INPUT"WHAT IS THE SUM OF THE THREE
   DIGITS ?";N
60 Q=37*N
70 PRINT:PRINT CHR$(7)
80 PRINT"YOUR ORIGINAL NUMBER WAS ";Q
90 PRINT:PRINT:GOTO 20
```

47 Gee Whiz V: Funny Similes

Give these newfangled gadgets an inch and they'll take a mile. In the case of your APPLE, give it some tacky retorts and it will spew out an endless string of dumb remarks.

You first tell the computer whom it is describing. Then it starts depicting that person in a series of colorful references.

Your non-computer friend won't understand it but you'll know the fun is in having the computer randomly select various words from a vocabulary you have given it. It combines those words into silly sayings.

In this program, the adjectives are in DATA lines 20-60. The nouns are in lines 70-110. Line 210 obtains the name of one of your "friends." Lines 300-350 randomly select an adjective. Lines 400-450 select a noun.

Line 500 matches your friend's name with the adjective and noun to create an insult. Line 520 makes it repeat continuously until you press the RESET key.

Program Listing

```
 10 HOME
 20 DATA SHORT,TALL,FAT,LEAN CLEAN
 30 DATA DIRTY,GOOD,BAD,HAPPY,SAD
 40 DATA GREEN,RED,YELLOW,BLUE,UGLY
 50 DATA PRETTY,SHARP,DULL,TACKY,NATTY
 60 DATA STRONG,WEAK,MEAN,NICE,DUMB
 70 DATA GNOME,TREE,PIG,BOX,CLOCK
 80 DATA TURKEY,GOLD,APPLE,DOG,ROOKIE
 90 DATA BEET,BIRD,SKY,SIN,PEACH
100 DATA TACK,RAZOR,PIN,PLUG,BULL
110 DATA WORM,LION,LAMB,PUPPY,OX
200 PRINT CHR$(7)
210 INPUT "WHOM ARE WE DESCRIBING ?";B$
300 T=INT(100*(RND(1)))
310 IF T<1 OR T>25 THEN GOTO 300
```

```
320 FOR L=1 TO T
330 READ D$
340 NEXT L
350 RESTORE
400 T=INT(100*(RND(1)))
410 IF T<26 OR T>50 THEN GOTO 400
420 FOR L=1 TO T
430 READ E$
440 NEXT L
450 RESTORE
500 PRINT CHR$(7)
510 PRINT B$;" IS ";D$;" AS A ";E$
520 GOTO 300
```

48 Gee Whiz VI: Who Is Youngest?

The computer asks for the names and ages of all people in the room. If you are the only person in the room, it will tell you to get someone else to play with you or forget it. After collecting the names and ages, the computer instantly tells which person is youngest.

Program Listing

```
 10 HOME
 20 PRINT"HI":PRINT"I'M YOUR APPLE
    COMPUTER"
 30 INPUT"WHAT'S YOUR NAME ";N$
 40 IF N$="" THEN PRINT"YOU HAVE TO
    HAVE A NAME":GOTO 30
 50 PRINT:PRINT"HI, ";N$
 60 INPUT"HOW MANY PEOPLE ARE IN THE
    ROOM ? ";Q
 70 IF Q<1 OR Q<>INT(Q) THEN 400
 80 IF Q=1 THEN 500
100 PRINT:PRINT"I BET I CAN TELL WHO
    IS YOUNGEST"
```

```
110 DIM P$(Q),A(Q)
120 FOR L=1 TO Q
130 IF L>1 THEN 170
140 INPUT"GIVE ME SOMEONE'S NAME "
    ;P$(L)
150 IF P$(L)="" THEN 140
160 IF L=1  THEN 190
170 PRINT:INPUT"GIVE ME ANOTHER NAME "
    ;P$(L)
180 IF P$(L)="" THEN 170
190 PRINT"HOW OLD IS ";P$(L)
200 INPUT A(L)
210 IF L=1 THEN LA=A(L):LN$=P$(L)
    :HA=A(L):HN$=P$(L)
220 IF A(L)<(LA) THEN LA=A(L):LN$=P$(L)
230 IF A(L)>(HA) THEN HA=A(L):HN$=P$(L)
240 NEXT L
250 HOME
260 PRINT N$;", HERE'S WHAT I FIND:"
270 PRINT:PRINT HN$;" IS OLDEST AT ";HA
280 PRINT"AND ";LN$;" IS YOUNGEST AT ";LA
290 PRINT:PRINT"THANKS FOR STOPPING BY,"
300 FOR L=1 TO Q
310 IF L=Q THEN 340
320 PRINT P$(L);","
330 GOTO 350
340 PRINT"AND ";P$(L)
350 NEXT L
360 END
400 PRINT:PRINT"DON'T PLAY AROUND, ";N$
410 PRINT"THERE CAN'T BE ";Q;" PEOPLE"
420 PRINT"PLEASE TELL THE TRUTH, ";N$
430 PRINT:GOTO 60
500 PRINT:PRINT"WELL, ";N$
510 PRINT"SINCE WE'RE ALL ALONE"
520 PRINT"THERE IS NO ONE TO DEMONSTRATE
    TO"
530 PRINT"SO BRING SOMEONE INTO THIS ROOM"
540 PRINT"OR FORGET IT"
550 PRINT:PRINT"WHICH WILL IT BE"
560 INPUT"GET OR FORGET ? ";G$
```

```
570 IF LEFT$(G$,1)="G" THEN PRINT
    :PRINT "OKAY":GOTO 60
580 IF LEFT$(G$,1)="F" THEN PRINT
    :PRINT "OKAY, GOODBYE":END
590 GOTO 560
```

Number Crunching

49 Number Reverser I

This program generates a series of random numbers and then adds them up. Then it reverses the order of the digits in the total. Creates a very strange random number!

Lines 20-40 generate three random single-digit numbers. Line 50 is a trap to make sure they are between 1 and 9 with no zeros showing up.

Line 60 changes the number values into strings so they can be tied together as three digits at line 70. Line 80 takes those three-digit strings from line 70 and turns them back into numerical values.

Line 110 is a time delay loop to stall a moment before line 120 pushes action back up to line 20 for a new start.

Program Listing

```
 10 HOME
 20 J=INT(10*(RND(1)))
 30 K=INT(10*(RND(1)))
 40 L=INT(10*(RND(1)))
 50 IF J<1 OR K<1 OR L<1 THEN GOTO 20
 60 J$=STR$(J):K$=STR$(K):L$=STR$(L)
 70 N$=J$+K$+L$:R$=L$+K$+J$
 80 N=VAL(N$):R=VAL(R$)
 90 PRINT"RANDOM NUMBER IS ";N
100 PRINT"BACKWARD, IT IS  ";R
110 FOR X=1 TO 500:NEXT X
120 PRINT:GOTO 20
```

50 Logic Functions

You can make your computer do things based on its decision that something exists. That is, in the first program listing here, it only will print the value of C if it finds that B has an existing value. If B is found to have no value, does not exist, C will not be printed.

The decision is in line 40. The machine only prints C if B does not equal zero. Since, in line 20, we set B = 10, the computer will find that something exists in B and, thus, go ahead and do the work assigned in the last half of line 40. If nothing had been stored in B, the last half of line 40 would have been ignored.

Program Listing

```
10 HOME
20 B=10
30 C=10*B
40 IF B THEN PRINT C
```

In the second program here, the APPLE only displays the results of the tests in lines 40 and 50 if the results of one or both is "true."

By doing the simple math in your head, you can see that the information in the right-hand side of line 20 is true. The information in the right-hand side of line 30 is false.

Line 20 says that 6 + 8 is greater than 3 times 4. That is, 14 is greater than 12. That is true.

Line 30 says that 5 + 2 is greater than 9 + 2. That is, 7 is greater than 11. That is false.

After reading line 20, the computer will store a 1 in B since the statement is true. Upon reading line 30, the computer will store a zero in C since the statement is false.

As action drops to line 40, the computer will find the 1 it stored in B and, thus, complete the action called for at the right-hand end of line 40. It will display the message, "B OKAY."

At line 50, however, the computer will find "nothing" (zero) in C and will not complete the right-hand end of that instruction. It only will do the right-hand end if it finds something in the left-hand end.

These logic functions are great for quick tests.

Program Listing

```
10 HOME
20 B=(6+8)>(3*4)
```

```
30 C=(5+2)>(9+2)
40 IF B THEN PRINT "B OKAY"
50 IF C THEN PRINT "C OKAY"
```

51 Factoring

This program finds and lists the factors of any number you specify, up to 10001. It can be used as a subroutine in a larger program, with appropriate attention to line numbers, variable names, and RETURN.

The number of individual factors are limited by the DIM statement in line 20. If you have less than the 48K RAM memory in our program-testing machine, try a smaller number such as 999 in line 20.

The list will exclude the number itself divided by 1.

For a quick sample run, try the number 18. You should find factors are 9, 6, 3 and 2. Or try 10001. Factors are 137 and 73.

Program Listing

```
 10 HOME
 20 DIM Q(5000)
 30 INPUT"NUMBER = ";N
 40 FOR L=2 TO N/2
 50 M=N/L
 60 IF M=INT(M) THEN P=P+1:Q(L)=M
 70 NEXT L
 80 PRINT"FACTORS ARE:"
 90 FOR L=1 TO N/2
100 IF Q(L)>1 THEN PRINT CHR$(7);Q(L)
    :GOTO 120
110 Z=Z+1
120 NEXT L
130 IF N=1 THEN PRINT CHR$(7);"NONE"
    :GOTO 150
140 IF Z=INT(N/2) THEN PRINT CHR$(7);"NONE"
150 PRINT:PRINT:CLEAR:GOTO 20
```

52 Standard Deviation

Here's a way to determine mean and standard deviation. In this particular program, you exit the entry cycle by entering the large number 999999999 (nine 9's) so you can't use 999999999 as one of your data points.

This is a great opportunity to experiment with standard deviation computations. Try a series of data points such as 3, 5, 3, 7, and 4. They should result in a total of 22; mean of 4.4; variance of 2.3999998; and standard deviation of 1.49666295.

Program Listing

```
 10 HOME
 20 INPUT"DATA POINT = ";X
 30 IF X=999999999 THEN 60
 40 T=T+X:S=S+X^2:N=N+1
 50 GOTO 20
 60 A=T/N:V=S/N-A^2:D=SQR(V)
 70 PRINT:PRINT CHR$(7)
 80 PRINT"DATA POINTS TOTAL: ";T
 90 PRINT"MEAN: ";A
100 PRINT"VARIANCE: ";V
110 PRINT"STANDARD DEVIATION: ";D
120 PRINT:PRINT:CLEAR:GOTO 20
```

53 Reciprocals

Key in any number. The computer will display its reciprocal. The actual conversion is done here at line 30.

Program Listing

```
10 HOME
20 INPUT"NUMBER TO CONVERT TO RECIPROCAL
   = ";N
30 R=1/N
40 PRINT"RECIPROCAL = ";R
50 PRINT:CLEAR:GOTO 20
```

54 Percentages

Usually it's more convenient to enter percentages as percent rather than having to convert to decimals in your head first. Of course, the computer needs that converted decimal value to do its work. How to get it?

This program does the trick. You give it a percentage and it converts that to a decimal. The computer does the hard work for you!

Line 30 makes the actual conversion. Use this idea as part of a larger check-balancing, accounting or bookkeeping program and save lots of mental effort.

Program Listing

```
10 HOME
20 INPUT"PERCENTAGE = ";P
30 D=.01*P
40 PRINT"DECIMAL = ";D
50 PRINT:CLEAR:GOTO 20
```

55 Above & Below A Line

Here's a way to count numbers above and below a cut-off line. The computer solicits numbers between 1 and 100. Any numbers you key in which are below 1 or above 100 are trapped out by line 140. Entering a zero ends the input cycle.

Line 50 counts the total numbers. Line 60 counts only those numbers between 1 and 50. Line 80 counts the numbers from 51 to 100. Lines 90 to 130 present results.

Program Listing

```
10 HOME:CLEAR
20 INPUT"GIVE ME A NUMBER";Z
30 IF Z=Ø THEN 80
40 IF Z<1 OR Z>100 THEN 20
```

```
 50 N=N+1
 60 IF Z<51 THEN B=B+1
 70 GOTO 20
 80 A=N-B
 90 PRINT:PRINT CHR$(7)
100 PRINT N;" NUMBERS TOTAL"
110 PRINT B;" 1-50"
120 PRINT A;" 51-100"
130 PRINT:PRINT:CLEAR:GOTO 20
```

56 Which Is Smallest?

How can the computer tell which number is smaller or larger? Here's how.

Type in the program and RUN it. It will ask for, and accept a continuous string of numbers until you end the input routine by keying in a zero.

Lines 40 to 60 make the decision as to which number is lowest.

Program Listing

```
 10 HOME:CLEAR
 20 INPUT"GIVE ME A NUMBER";Z
 30 IF Z=Ø THEN 80
 40 N=N+1
 50 IF N=1 THEN D=Z
 60 IF Z<D THEN D=Z
 70 GOTO 20
 80 PRINT:PRINT CHR$(7)
 90 PRINT"THE SMALLEST NUMBER WAS ";D
100 PRINT:PRINT:CLEAR:GOTO 20
```

57 Two-Digit Round Off

It is possible to round off to the nearest hundredths place. That is, to two digits after the decimal point. Here's how:

Program Listing

```
10 HOME:CLEAR
20 PRINT"GIVE ME A NUMBER TO"
30 PRINT"MORE THAN TWO DECIMAL PLACES"
40 INPUT"ORIGINAL AMOUNT $";N
50 R=INT(100*N+.5)/100
60 PRINT CHR$(7)
70 PRINT "$";N;" ROUNDS OFF TO $";R
80 PRINT:CLEAR:GOTO 20
```

58 Dump The Integer

Look at the number 123.456 with an eye toward how to get rid of the portion left of the decimal point. Keep only .456 and dump 123. Here's a short program to accomplish that.

Try 5.67. It will come out .67. Or 500.5 which will come out .5.

Caution: your APPLE is *not* always accurate at math. Try 987.65, for example. It will result in .650000095 because when the APPLE subtracts 987 from 987.65 at line 30 it does not get .65 as the correct answer. Rather, it gets an approximate answer!

Program Listing

```
10 HOME:CLEAR
20 INPUT"GIVE ME A NUMBER WITH A
   DECIMAL";N
30 X=N-INT(N)
```

```
40 PRINT CHR$(7)
50 PRINT X
60 PRINT:CLEAR:GOTO 20
```

59 Random Sampler

This program strengthens your confidence in the random number generator built into your APPLE computer.

It generates 100 numbers between zero and 100 and tells you how many of those are above 49 and how many are below 50. See the sample RUN for several sets of results in our recent test.

Program Listing

```
 10 HOME
 20 FOR L=1 TO 100
 30 X=INT(100*(RND(1)))
 40 IF X<50 THEN Y=Y+1
 50 IF X>49 THEN N=N+1
 60 NEXT L
 70 PRINT Y;" YES"
 80 PRINT N;" NO"
 90 N=Ø:Y=Ø
100 PRINT:GOTO 20
```

Sample Run

```
 51  YES          46  YES
 49  NO           54  NO

 42  YES          52  YES
 58  NO           48  NO

 56  YES          48  YES
 44  NO           52  NO
```

60 Averages

Key in numbers in any order. A zero will end entry. The computer will tell you the average number of all numbers you entered.

Line 40 finds the total number of all numbers entered. Line 50 finds the total of entered numbers. Line 70 computes the average.

Program Listing

```
 10 HOME:CLEAR
 20 INPUT"GIVE ME A NUMBER";Z
 30 IF Z=Ø THEN 70
 40 N=N+1
 50 T=T+Z
 60 GOTO 20
 70 A=T/N
100 PRINT:PRINT CHR$(7)
110 PRINT"THE AVERAGE NUMBER IS ";A
120 PRINT:PRINT:CLEAR:GOTO 20
```

61 Mid-Range Number

Here's how to find the middle of a range of numbers. You key in as many numbers in a series as you wish. After the last number, key in a zero to move the program out of the entry cycle.

Lines 40 to 70 select the highest and lowest numbers in the range. They actually define the range. Then line 90 finds the middle point of that range.

Program Listing

```
10 HOME:CLEAR
20 INPUT"GIVE ME A NUMBER";Z
30 IF Z=Ø THEN 90
```

```
 40 N=N+1
 50 IF N=1 THEN H=Z:L=Z
 60 IF Z<L THEN L=Z
 70 IF Z>H THEN H=Z
 80 GOTO 20
 90 M=L+((H-L)/2)
100 PRINT:PRINT CHR$(7)
110 PRINT"THE MID-RANGE NUMBER IS ";M
120 PRINT:PRINT:CLEAR:GOTO 20
```

62 Which Is Largest?

Suppose you have a group of numbers and you would like to know which number is largest within the group? Here's a software routine for your APPLE so it can locate the largest number.

You can key in as many numbers as you wish. To end that entry cycle, type in a zero. The computer will see that zero as its cue to leave the entering routine and get on with computing.

Line 40 tests each new number as you enter it. If a new number is larger, that new number is stored in memory location D. At the end of the entry cycle, the largest number is left stored in D. Line 70 recalls that largest number and prints it.

Program Listing

```
10 HOME:CLEAR
20 INPUT"GIVE ME A NUMBER";Z
30 IF Z=Ø THEN 60
40 IF Z>D THEN D=Z
50 GOTO 20
60 PRINT:PRINT CHR$(7)
70 PRINT"THE LARGEST NUMBER WAS ";D
80 PRINT:PRINT:CLEAR:GOTO 20
```

63 Percent to Decimal

Checking, interest, sales tax, and other financial programs are more "user friendly" if you don't have to make manual conversions in your head. For example, if you know your savings account earns 8 percent interest, and you need to multiply by the decimal value for 8 percent (which is 0.08), it is easier to be able to enter 8 and let the APPLE figure out the decimal value.

Here's another way to change percentages to decimals inside a program to simplify entry by permitting percents to be entered as simple numbers.

For some examples, try entering a price of 2.50 and a sales tax percentage of 6. Your APPLE will find the bill totals $2.65. Or try $7.80 and 5 percent tax. The bill will be $8.19. Try $123.75 at 8 percent tax. The bill will total $133.65.

Program Listing

```
10 HOME:CLEAR
20 INPUT"PRICE = $";P
30 INPUT"SALES TAX PERCENTAGE = ";R
40 T=.01*R
50 B=P+T*P
60 PRINT:PRINT CHR$(7)
70 PRINT"BILL TOTALS $";B
80 PRINT:PRINT:GOTO 20
```

64 Rounding Off

The techniques for rounding off numbers is easy. This program, which can stand alone or be worked into a larger program as a subroutine, rounds a decimal to the nearest whole number.

There are two views on how to round off. One holds

that "if the number is more than five, you round up." Which means that exactly 0.5 rounds down.

Another view is that "any number less than five rounds down." In that case exactly 0.5 rounds up.

The first set of program lines below is for the fellow with the "more than five rounds up" idea.

Program Listing

```
 10 HOME:CLEAR
 20 INPUT"GIVE ME A NUMBER TO BE ROUNDED
    OFF";N
 30 IF N>INT(N) THEN 50
 40 R=N:GOTO 100
 50 D=N-INT(N)
 60 IF D>0.5 THEN 90
 70 R=INT(N)
 80 GOTO 100
 90 R=INT(N)+1
100 PRINT CHR$(7)
110 PRINT N;" ROUNDS OFF TO ";R
120 PRINT:CLEAR:GOTO 20
```

The second set of program lines rounds off on the "less than five rounds down" theory. Line 50 finds the decimal portion of the number and lines 60-90 do the rounding.

Program Listing

```
 10 HOME:CLEAR
 20 INPUT"GIVE ME A NUMBER TO BE ROUNDED
    OFF";N
 30 IF N>INT(N) THEN 50
 40 R=N:GOTO 100
 50 D=N-INT(N)
 60 IF D<0.5 THEN 90
 70 R=INT(N)+1
 80 GOTO 100
 90 R=INT(N)
100 PRINT CHR$(7)
110 PRINT N;" ROUNDS OFF TO ";R
120 PRINT:CLEAR:GOTO 20
```

65 Number Reverser II

Give the computer a number. It will immediately, display the number printed backward on the video display.

Line 20 collects the number from you. Line 30 measures its length. Lines 40-70 reverse it and line 80 displays the reversed number.

Program Listing

```
10 HOME
20 INPUT"GIVE ME A NUMBER ";N$
30 L=LEN(N$)
40 FOR Y=L+1 TO 1 STEP -1
50 B$=B$+MID$(N$,Y,1)
60 NEXT Y
70 B=VAL(B$)
80 PRINT"REVERSED, THE NUMBER IS ";B
90 PRINT:CLEAR:GOTO 30
```

66 Find Highest/Lowest

Suppose we have a list of people and each person has been assigned a number or score. This program accepts the names and scores and sorts out the persons with the highest and lowest scores. Here's how it works.

At line 20, the memory is DIMensioned to hold data on 100 persons. Lines 30-130 take in the info on each person. As each person's score is entered, lines 100-120 determine if it is higher or lower than all previous scores. If higher or lower, it is so noted.

To complete data entry, simply press ENTER without data. That will prompt the computer, at lines 140 and 150, to print the lowest score and the highest score.

Program Listing

```
 10 HOME
 20 DIM M$(100)
 30 FOR L=1 TO 100
 40 INPUT"NAME: ";MN$
 50 IF MN$="" THEN 140
 60 INPUT"SCORE: ";KW$
 70 M$(L)=MN$+" "+KW$
 80 N=N+1
 90 K=VAL(KW$)
100 IF N=1 THEN LL=K:LM$=M$(L)
    :HH=K:HM$=M$(L)
110 IF K<LL THEN LL=K:LM$=M$(L)
120 IF K>HH THEN HH=K:HM$=M$(L)
130 NEXT L
140 PRINT"LOWEST SCORE: ";LM$
150 PRINT"HIGHEST SCORE: ";HM$
```

Sample Run

```
RUN RETURN
NAME:
FRED RETURN
SCORE:
45  RETURN

NAME:
HELEN  RETURN
SCORE:
88 RETURN

NAME:
JOHN RETURN
SCORE:
76 RETURN

NAME: RETURN
WAYNE
SCORE:
62  RETURN
```

```
NAME:
GEORGE RETURN
SCORE
51 RETURN

NAME:
RETURN
LOWEST SCORE: FRED 45
HIGHEST SCORE: HELEN 88
```

67 Every 10th Answer

This program generates a random number in the range of zero to 999. However, it has a difference. It only shows you every tenth number it generates.

Line 20 generates the numbers. Line 40 selects the tenth number from each set.

Program Listing

```
10 HOME
20 T=INT(1000*(RND(1)))
30 V=V+1
40 IF .1*V=INT(.1*V) THEN PRINT V,T
50 GOTO 20
```

68 Random Numbers: Zero to Nine

Although you see four program lines below, what we really have here is a very convenient single-line program for you to insert in a larger game or educational-testing program.

Line 20 is the winner here. It prints a random number from zero to nine every time. For your use here, we print

that number on the screen. You could just as easily have the computer store that random number in a memory location for later recall and use.

We have added lines 10, 30 and 40 to make your APPLE show you a whole series of random numbers from zero to nine. Remember, line 20 is the important single-line program element here.

If you would like random numbers in the range from zero to 99, make it 100* in line 20. For zero to 999, use 1000* in line 20.

Program Listing

```
10 HOME
20 PRINT INT(10*RND(1))
30 FOR L=1 TO 200:NEXT L
40 GOTO 20
```

69 Random Numbers: Distribution

Ever wonder how "random" are the numbers generated by the random-number generator in your APPLE when you use the RND instruction? Try this program.

It generates 100 random numbers in a range from zero to nine and counts how many there are of each number between zero and nine.

By the way, while it is doing that it will display the message "counting" so you can tell it is working.

At the end of its run, the APPLE prints a neat chart, on the video display, of results. See our sample run.

Program Listing

```
10 HOME
20 FOR L=1 TO 100
30 N=INT(10*RND(1))
```

```
 40 IF N=Ø THEN A=A+1
 50 IF N=1 THEN B=B+1
 60 IF N=2 THEN C=C+1
 70 IF N=3 THEN D=D+1
 80 IF N=4 THEN E=E+1
 90 IF N=5 THEN F=F+1
100 IF N=6 THEN G=G+1
110 IF N=7 THEN H=H+1
120 IF N=8 THEN I=I+1
130 IF N=9 THEN J=J+1
140 PRINT "COUNTING"
150 NEXT L
160 HOME
170 PRINT"Ø: ";A
180 PRINT"1: ";B
190 PRINT"2: ";C
200 PRINT"3: ";D
210 PRINT"4: ";E
220 PRINT"5: ";F
230 PRINT"6: ";G
240 PRINT"7: ";H
250 PRINT"8: ";I
260 PRINT"9: ";J
270 END
```

Sample Run

```
RUN RETURN

COUNTING

0:  8
1: 10
2: 14
3: 16
4:  6
5:  7
6:  7
7: 15
8:  4
9: 13
```

```
RUN  RETURN

COUNTING

0:  6
1: 12
2: 11
3: 13
4:  7
5: 12
6:  9
7: 10
8:  8
9: 12
```

70 Random Numbers: Averages

This program generates 100 random numbers and totals them. Then it finds the average of all 100 numbers.

In fact, the average number itself is a useful new random number.

Program Listing

```
 10 HOME
 20 FOR L=Ø TO 99
 30 N=INT(10*RND(1))
 40 NT=NT+1
 50 PRINT"AVERAGING"
 60 NEXT L
 70 HOME
 80 AV=NT/100
 90 PRINT"TOTAL OF ALL RANDOM NUMBERS
    IS ";NT
100 PRINT
110 PRINT"AVERAGE NUMBER IS ";AV
120 END
```

Sample Run

```
RUN  RETURN

AVERAGING

TOTAL OF ALL RANDOM NUMBERS IS 444
AVERAGE NUMBER IS 4.44

RUN  RETURN

AVERAGING

TOTAL OF ALL RANDOM NUMBERS IS 424
AVERAGE NUMBER IS 4.24
```

71 Random Numbers: Sorting Hi/Lo

It's important to be able to sort a group of numbers to see what the highest and lowest values are. This program does that.

The random number generator is in line 30. It gives numbers in a range of zero to 999. Line 50 determines the lowest number in the set and line 60 finds the highest number.

Program Listing

```
 10 HOME
 20 FOR L=Ø TO 99
 30 N=INT(1000*RND(1))
 40 IF L=Ø THEN LN=N:HN=N
 50 IF N<LN THEN LN=N
 60 IF N>HN THEN HN=N
 70 PRINT"SORTING"
 80 NEXT L
100 HOME
```

```
110 PRINT"LOW NUMBER IS ";LN
120 PRINT
130 PRINT"HIGH NUMBER IS ";HN
300 END
```

Sample Run

```
RUN  RETURN

SORTING

LOW NUMBERS IS 32
HIGH NUMBER IS 983

LOW NUMBER IS 14
HIGH NUMBER IS 980

LOW NUMBER IS 17
HIGH NUMBER IS 985

LOW NUMBER IS 9
HIGH NUMBER IS 991

LOW NUMBER IS 1
HIGH NUMBER IS 994
```

72 Number Reverser III

Give your APPLE any three-digit number and, as a result of this particular programming trick, it will reverse the original number. For example, 789 will be transformed into 987. Or 123 into 321. It takes your three-digit number apart and reassembles it in reverse order.

Be careful! Sometimes your APPLE won't be too accurate. Try the number 234. It will reverse that to 332. The problem is, when APPLE subtracts 23 from 23.4 at line 60, it gets an approximate answer .399999999 rather than .4. So the digit you know should be a 4 gets translated as a 3.

Program Listing

```
10 HOME:CLEAR
20 INPUT"GIVE ME A THREE-DIGIT NUMBER";N
30 IF N<100 OR N>999 THEN 20
40 A=INT(N/100)
50 B=INT(10*((N/100)-A))
60 C=INT(10*((N/10)-(INT(N/10))))
70 PRINT CHR$(7)
80 PRINT C;B;A
90 PRINT:CLEAR:GOTO 20
```

Business Matters

73 Mark Up

Mr. Storekeeper, here's just what you have needed to compute mark ups. This program causes your APPLE to find the retail price for which your percentage off would give the wholesale cost.

For instance, if you got 40 percent off on an item and paid $60, how much was it priced at, at retail? The answer is $100. To put that another way, if retail price or suggested retail price is $100 and you got 40 percent off at wholesale, what is the wholesale price? The answer is $60.

Try $40 wholesale which is 60 percent off. The answer is $100 retail. Or try $10 wholesale at 90 percent off. Retail would be $100. Or $75 wholesale at 25 percent off gives $100 retail.

Here's a toughie! Try $19.95 wholesale cost. Mark-up percentage is 40. The correct retail answer is $33.25.

Program Listing

```
10 HOME
20 INPUT"WHOLESALE COST = $";W
30 INPUT"MARK-UP PERCENTAGE = ";P
40 D=1-.01*P:R=W/D
50 PRINT"RETAIL PRICE = $";R
60 PRINT:CLEAR:GOTO 20
```

74 Percentage Off

From earlier tips in this book, you know how to make your APPLE convert percentages to decimals. But what if you want to know "percentage off?"

For example, how much is 40 percent off? This program can be used to interpret 40 percent off and compute the decimal value needed. Try 40 percent off $100. The computer will change 40 percent off into decimal value

0.60. If you multiply 0.60 times $100 you find $60 is 40 percent off $100.

Line 30 makes the important translation.

Program Listing

```
10 HOME
20 INPUT"PERCENTAGE OFF = ";P
30 D=1-.01*P
40 PRINT"DECIMAL = ";D
50 PRINT:CLEAR:GOTO 20
```

75 To Nearest 95¢

Many companies like to price their goods at a figure ending in 95 cents. For instance, a ten dollar item might be marked $9.95 or $10.95.

Here's a program which demonstrates how to make all prices come out to the nearest 95 cents. See line 40. It merely takes the integer portion of the dollars number and adds 0.95 to it.

Program Listing

```
10 HOME:CLEAR
20 INPUT"MANUFACTURING COST = $";C
30 INPUT"PRICING MULTIPLIER = ";M
40 P=INT(C*M)+.95
50 PRINT CHR$(7)
60 PRINT:PRINT"RETAIL PRICE = $";P
70 PRINT:PRINT:GOTO 20
```

Sample Run

```
MFG COST = $
1.12  RETURN
PRICING MULTIPLIER =
10  RETURN
RETAIL PRICE = $11.95
RETURN
```

```
MFG COST = $
.79 RETURN
PRICING MULTIPLIER =
8  RETURN
RETAIL PRICE = $6.95
RETURN

MFG COST = $
123.45 RETURN
PRICING MULTIPLIER =
3  RETURN
RETAIL PRICE = $370.95
```

76 To The Nearest Penny

This program is useful when you have a dollar-and-cents figure with more than two decimal places. For example, $151.6972. You need to transform $151.6972 to the more common $151.70

This small program would make a good subroutine in a larger set of instructions. To do so, insert GOSUB at the appropriate place in the larger set of program lines. Modify the line numbers of this small program so the subroutine will be located in an unused position in the larger listing. Change the last line of this small program to RETURN. Delete the first line.

Program Listing

```
10 HOME:CLEAR
20 PRINT"ENTER A NUMBER TO"
30 PRINT"MORE THAN TWO DECIMAL PLACES"
40 PRINT:INPUT"ORIGINAL AMOUNT = $";N
50 R=INT(100*N+.5)/100
60 PRINT CHR$(7)
70 PRINT:PRINT"TO THE NEAREST PENNY,"
80 PRINT"$";N;" IS $";R
90 PRINT:PRINT:GOTO 40
```

77 Dollars & Cents

If the result of your computation is a "money" answer, and you don't know whether to display it in dollars or cents, let the computer decide.

This program decides whether to display the output in dollars or cents. Notice in the sample run that, in the first instance, 1000 @ $1350 became $1.35 while 1000 @ $18.95 became 1.89 cents. Line 50 in the program makes the decision.

Program Listing

```
10 HOME:CLEAR
20 INPUT"QUANTITY = ";P
30 INPUT"TOTAL COST = $";C
40 T=C/P
50 IF T<1 THEN T=100*T:GOTO 80
60 PRINT"EACH COST $";T
70 PRINT:GOTO 20
80 PRINT"EACH COST ";T;" CENTS"
90 PRINT:GOTO 20
```

78 Daily Code

Need to have a secret code each day of the year? This software generates a list of code numbers. Of course, you can change the list every day if you wish.

Program Listing

```
 10 HOME
 20 GOSUB 200
100 PRINT "SUNDAY:    ";C:GOSUB 200
110 PRINT "MONDAY:    ";C:GOSUB 200
120 PRINT "TUESDAY:   ";C:GOSUB 200
```

```
130 PRINT "WEDNESDAY: ";C:GOSUB 200
140 PRINT "THURSDAY:  ";C:GOSUB 200
150 PRINT "FRIDAY:    ";C:GOSUB 200
160 PRINT "SATURDAY:  ";C:END
200 C=INT(10000*(RND(1)))
210 IF C<1000 THEN GOTO 200
220 RETURN
```

79 One-Time Password

If you don't want unauthorized use of your programs, insert a requirement that a user know a password. This particular routine allows only one try at entering a correct password.

For our password, we have selected "elephant" and stored it in line 30. You can change the password to whatever you like.

If a correct attempt at entering the password is made, program action will progress to line 100. Otherwise, action drops to line 40 and action ends.

Program Listing

```
 10 HOME
 20 INPUT"WHAT IS THE PASSWORD ?";A$
 30 IF A$="ELEPHANT" THEN GOTO 100
 40 END
100 PRINT"YOU GOT IT RIGHT"
110 PRINT"THE PROGRAM WILL RUN"
```

80 Three-Tries Password

Here the software lets you try three times to enter the correct password. You don't get to go forward with the program if you don't get it right in three tries.

Again the password is "elephant" and is stored in

line 30. You can change the password to whatever suits you.

Lines 40 to 60 allow the three attempts. If no good after three tries, then END.

Program Listing

```
 10 HOME
 20 INPUT"WHAT IS THE PASSWORD ?";A$
 30 IF A$="ELEPHANT" THEN GOTO 100
 40 B=B+1
 50 IF B=3 THEN END
 60 GOTO 20
100 PRINT"YOU GOT IT RIGHT"
110 PRINT"THE PROGRAM WILL RUN"
```

81 Multiple Passwords

Here's a really complex password entry system. It has a unique "account number" and a password for each person. This will allow several different persons access to the program but each person will have a different combination to the lock!

account number	password
12345	zebra
23456	goose
34567	trout
45678	snake

Each individual user must correctly enter his unique account number and then his own personal password. If account number is wrong, then the password never can be right. If account number is okay but password doesn't match, the user gets no run.

You can add users to this program by adding lines to the 300-340 subroutine.

Program Listing

```
 10 HOME:PRINT CHR$(7)
 20 FOR L=1 TO 3
 30 INPUT"YOUR ACCOUNT NUMBER ?";UA
 40 GOSUB 300
 50 INPUT"PASSWORD ?":PS$
 60 IF PS$=PW$ THEN 100
 70 NEXT L
 80 END
100 PRINT"MAIN PROGRAM RUNS"
110 END
300 IF UA=12345 THEN PW$="ZEBRA"
310 IF UA=23456 THEN PW$="GOOSE"
320 IF UA=34567 THEN PW$="TROUT"
330 IF UA=45678 THEN PW$="SNAKE"
340 RETURN
```

82 Yes/No Decision Maker

This is handy for the busy executive who doesn't have time for decisions.

Line 10 clears the screen. Line 20 generates a random number from zero to 100. Line 30 selects a *yes* answer if the random number is greater than 49. Otherwise, line 40 chooses a *no* answer.

Program Listing

```
10 HOME
20 X=100*(RND)1
30 IF X>49 THEN PRINT "YES":GOTO 50
40 PRINT "NO"
50 END
```

83 Wages & Hours

These useful lines compute total hours worked at regular pay and number of hours worked at time-and-a-half overtime. The computer then finds gross pay and rounds off to the nearest cent.

The program knows that overtime starts after 40 hours. It makes payroll bookkeeping quick and simple.

Program Listing

```
 10 HOME
 20 INPUT"HOURLY PAY RATE = $";P
 30 INPUT"NUMBER HOURS WORKED = ";H
 40 IF H>40 THEN OT=H-40:GOTO 100
 50 W=H*P
 60 PRINT"GROSS WAGES = $";W
 70 END
100 W=(40*P)+(OT*P*1.5)
110 PRINT"GROSS WAGES = $";W
120 END
```

Sample Run

```
RUN RETURN

HOURLY PAY RATE = $
5.75 RETURN
NUMBER HOURS WORKED =
61 RETURN
GROSS WAGES = $411.125
```

84 Invoicing

There's a lot of repetitious math work to be done before you mail invoices to your customers. This software has the computer collect a few pertinent bits of data from you

and then present all the various totals you need to plug into an invoice.

It gives you a total retail price for all goods sold on the invoice, total sales tax if applicable, shipping charges and the grand total amount due you from your customer.

Program Listing

```
 10 HOME
 20 INPUT"QUANTITY SOLD = ";Q
 30 INPUT"UNIT PRICE = $";P
 40 INPUT"SALES TAX RATE PERCENT = ";S
 50 INPUT"SHIPPING CHARGES = $";H
 60 S=.01*S:C=Q*P:T=C*S:F=C+T+H
 70 CC=INT(100*C+.5)/100
 80 TT=INT(100*T+.5)/100
 90 FF=INT(100*F+.5)/100
100 HOME
110 PRINT"TOTAL PRICE = $";CC
120 PRINT"SALES TAX = $";TT
130 PRINT"SHIPPING CHARGES = $";H
140 PRINT
150 PRINT"INVOICE TOTAL = $";FF
200 END
```

Sample Run

```
RUN  RETURN

QUANTITY SOLD
178  RETURN
UNIT PRICE = $
55.98 RETURN
SALES TAX RATE PERCENT =
6    RETURN
SHIPPING CHARGES = $
100  RETURN

TOTAL PRICE = $9964.44
SALES TAX = $597.87
SHIPPING CHARGES = $100

INVOICE TOTAL = $10662.31
```

85 Invoice Computer

The computer will ask you for discount percentage to be applied to the invoice; the retail price of goods being invoiced; and the quantity of those goods sold.

Then it will ask if you have other items on the same invoice and, if so, will again get price and quantity sold info from you. It will assume the same discount applies.

When you tell the computer there are no more items for the invoice, it will display the total.

Program Listing

```
 10 GOSUB 600
 20 PRINT:PRINT"INFORMATION FOR
    COMPUTING INVOICES:"
 30 PRINT:INPUT"WHAT IS DISCOUNT
    PERCENTAGE ";DP
 40 DD=1-.01*DP
 50 INPUT"WHAT IS THE ITEM SOLD ";IT$
 60 INPUT"WHAT IS RETAIL PRICE ";P
 70 INPUT"WHAT IS THE QUANTITY SOLD ";Q
 80 C=P*DD:I=C*Q:T=T+I
 90 PRINT:PRINT"WHOLESALE COST OF ";Q;
    " ";IT$;"S:  $";I
100 PRINT:PRINT:PRINT
110 PRINT"IF THERE ARE MORE ITEMS"
120 PRINT"FOR THE SAME INVOICE, PRESS M"
130 PRINT:PRINT"IF NO OTHER ITEMS,PRESS T"
140 Z=PEEK(-16384)
150 POKE -16368,Ø
160 IF Z<128 THEN 140
170 IF Z=205 THEN HOME:GOTO 50
180 IF Z=212 THEN 200
190 GOTO 140
200 PRINT:PRINT CHR$(7)
210 PRINT"INVOICE GRAND TOTAL:  $";T
220 PRINT:PRINT
230 PRINT"TO DO OTHER INVOICES"
```

```
240 PRINT"AT OTHER DISCOUNTS, PRESS
    ANY KEY"
250 GET KY$
260 CLEAR:GOTO 10
600 HOME
610 PRINT "**********************"
620 PRINT "* INVOICE COMPUTER *"
630 PRINT "**********************"
640 RETURN
```

86 Unit Price

Suppose you find 895 green Widgets and buy them for $695. How much did each green Widget cost? Rounded off, $7.77.

Unit price is total price divided by quantity. The quantity can be expressed in weight, total numbers, etc. It works the same whether you are talking about pounds of coffee, yards of concrete, gallons of ice cream, boxes of books, or units of Widgets.

This program asks for the name of the item, quantity purchased and total price paid. It then displays quantity, name, total and unit price.

Program Listing

```
 10 HOME
 20 INPUT"ITEM NAME IS ";N$
 30 INPUT"QUANTITY OF ITEMS = ";Q
 40 INPUT"TOTAL PRICE PAID FOR ITEMS =
    $";P
 50 U=P/Q
 60 FOR L=1 TO 10
 70 SOUND=PEEK(-16336)
 80 NEXT L
 90 HOME
100 PRINT N$;" UNIT PRICE = $";U
200 END
```

Sample Run

```
RUN  RETURN

ITEM NAME IS
WIDGETS RETURN
QUANTITY OF  ITEMS =
999 RETURN
TOTAL PRICE PAID FOR ITEMS = $
14653 RETURN

BEEP WIDGETS UNIT PRICE = $14.6676677
```

87 Sorting Inventory

Here's a time-saver, useful for small businesses in evaluating the contents of inventory. As written, it can handle 50 different inventory items. To change that quantity, increase or decrease the number 50 in lines 10 and 100 and the number 49 in line 430.

The computer will ask you the name of a product's manufacturer, what the product is, its retail price, and the quantity on hand in inventory.

It sorts all such information on the various items in inventory and lists, alphabetically by manufacturer, and then lists again, alphabetically by product name.

Note in the sample run, the program can tell the difference between a manufacturer's similar products and between the same products from different manufacturers. If one manufacturer has the same product available at different prices, the computer sorts by price. See the sample run for an example of this.

Use a printer and you will be able to print-out for your permanent records an alphabetized list of all products in inventory and a summary review of quantities, prices, and values.

Whether you display results on your television screen or on a printer, statistics include total number of different

items in inventory; number of different items in various price ranges; and total retail value of the inventory.

You can change the price ranges by modifying the less-than values in lines 170-210.

Program Listing

```
 10 HOME:DIM M$(50)
 20 PRINT"********************"
 30 PRINT"* INVENTORY REVIEW *"
 40 PRINT"********************"
 50 PRINT:PRINT:PRINT CHR$(7)
100 FOR L=1 TO 50
110 INPUT"MANUFACTURER:  ";P$
120 IF P$="" THEN 400
130 INPUT"PRODUCT:  ";Q$
140 INPUT"RETAIL PRICE:  $";G$
150 INPUT"QUANTITY ON HAND:  ";GG$
160 X=VAL(G$):XX=VAL(GG$)
170 IF X<10 THEN F=F+1:GOTO 220
180 IF X<20 THEN D=D+1:GOTO 220
190 IF X<30 THEN C=C+1:GOTO 220
200 IF X<40 THEN B=B+1:GOTO 220
210 A=A+1
220 Y=Y+X
230 Z=Z+1
240 IF Z=1 THEN J=X:K=X
250 IF X<J THEN J=X
260 IF X>K THEN K=X
270 I=I+X
280 II=XX*X:TT=TT+II
290 H=I/Z
300 M$(L)=P$+" "+Q$+" "+"$"+G$+" "+GG$
310 PRINT
320 NEXT L
400 HOME:PRINT CHR$(7)
410 PRINT"SORTING"
420 T=Ø
430 FOR L=1 TO 49
440 IF M$(L)<=M$(L+1) THEN 460
450 E$=M$(L):M$(L)=M$(L+1):M$(L+1)=E$:T=1
460 NEXT L
```

```
470 IF T=1 THEN 420
500 HOME:PRINT CHR$(7)
510 PRINT"MFG / ITEM / PRICE / QUANTITY"
520 PRINT
530 FOR L=1 TO 50
540 IF M$(L)<>"" THEN PRINT M$(L)
550 NEXT L
560 PRINT:PRINT
570 PRINT"TO REVIEW INVENTORY, PRESS ANY KEY"
580 GET KY$
700 HOME:PRINT CHR$(7)
710 PRINT TAB(10)"INVENTORY REVIEW"
720 PRINT:PRINT"NUMBER DIFFERENT ITEMS:"
730 PRINT A;" FROM $40.00 ON UP"
740 PRINT B;" FROM $30.00 TO 39.99"
750 PRINT C;" FROM $20.00 TO 29.99"
760 PRINT D;" FROM $10.00 TO 19.99"
770 PRINT F;" UNDER $10.00"
780 PRINT:PRINT"TOTAL ";Z;" ITEMS"
790 PRINT:PRINT"AVERAGE PRICE $";H
800 PRINT"RANGE $";J;" TO $";K
810 PRINT"TOTAL RETAIL VALUE $";TT
820 PRINT:PRINT:PRINT
830 PRINT"TO DO ANOTHER SET, PRESS ANY KEY"
840 GET KY$
850 CLEAR:GOTO 10
```

Sample Run

```
**********************
* INVENTORY REVIEW *
**********************

MANUFACTURER:
BROWN    RETURN
PRODUCT:
WIDGET   RETURN
RETAIL PRICE:  $
9.95     RETURN
QUANTITY ON HAND:
100      RETURN
```

```
MANUFACTURER:
SMITH   RETURN
PRODUCT:
GLOVES  RETURN
RETAIL PRICE:  $
12.99   RETURN
QUANTITY ON HAND:
40      RETURN

MANUFACTURER:
JONES   RETURN
PRODUCT:
OVENS   RETURN
RETAIL PRICE:  $
39.99   RETURN
QUANTITY ON HAND:
17      RETURN

MANUFACTURER:
        RETURN

SORTING

MFG / ITEM / PRICE / QUANTITY

BROWN WIDGET $9.95 100
JONES OVENS $39.99  17
SMITH GLOVES $12.99 40

TO REVIEW INVENTORY, PRESS ANY KEY
H

    INVENTORY REVIEW
NUMBER DIFFERENT ITEMS:
Ø FROM $40.00 ON UP
1 FROM $30.00 TO 39.99
Ø FROM $20.00 TO 29.99
1 FROM $10.00 TO 19.99
1 UNDER $10.00


TOTAL 3 ITEMS
```

```
AVERAGE PRICE $20.9766667
RANGE $9.95 TO $39.99
TOTAL RETAIL VALUE $2194.43

TO DO ANOTHER SET,PRESS ANY KEY
```

Graphics

88 Unending List

It may be useless but it sure is a neat visual effect! Looks very important. Provides an eye-catching display of continually-rolling words.

It will work with just the one program line but looks a lot better when you have many program lines in the computer. Try putting it at the end of some other program.

```
10000 LIST:GOTO 10000
```

To slow down the listing action scroll, try:

```
10000  LIST
10010  FOR L = 1 TO 100:NEXT L
10020  GOTO 10000
```

89 Drawing Sketches

Now you can draw lines, rules, diagrams, maps, charts, boxes—anything you can imagine—on the face of your video monitor. Use the APPLE keyboard as your pen and its video output as your ink.

Drawings appear in the upper half of the video screen. With the program running, type U to move the line upward; D for down; R for right; and L for left.

Be patient. The keyboard scan in line 20 doesn't always pick up your key press. You sometimes will have to press twice or three times. But you can draw mazes, names, letters, pictures.

These drawings, in the "text" mode, are much less defined than those might be in the graphic mode but useful since the machine remains in the text mode.

Program Listing

```
10 HOME:Z=Ø:X=Ø:Y=Ø
20 Z=PEEK (-16384)
30 POKE -16368,Ø
40 IF Z=196 OR Z=204 OR Z=210 OR Z=213
   THEN GOTO 60
```

```
 50 Z=Ø:GOTO 20
 60 IF Z=213 THEN Y=Y-1:GOTO 200
 70 IF Z=196 THEN Y=Y+1:GOTO 200
 80 IF Z=210 THEN X=X+1:GOTO 200
 90 IF Z=204 THEN X=X-1:GOTO 200
100 GOTO 20
200 COLOR=Ø
210 IF X>39 THEN X=39
220 IF X<Ø THEN X=Ø
230 IF Y>24 THEN Y=24
240 IF Y<Ø THEN Y=Ø
250 PLOT X,Y
260 Z=Ø
270 GOTO 20
```

90 Eyeball Scrambler

Blink. Blink. Blink. It's enough to make your eyeballs hurt!

The screen is filled with all the many printable keyboard characters. Some flash on and off. Some do not. Creates a very lively display!

Program Listing

```
10 HOME
20 FOR V=1024 TO 1999
30 R=INT(100*(RND(95)))
40 IF R<33 THEN GOTO 30
50 POKE V,R
60 NEXT V
70 GOTO 70
```

91 Flashing Dot

Want a blinking dot in the upper left-hand corner of the screen? You've got it!

This program clears the screen and causes that dot to blink. Timing loops in lines 20 and 40 control the length of time the dot is on and off.

See ASCII number list, in an earlier tip in this book, for other characters to display as a dot. We use 47 which appears as a "slash bar."

Program Listing

```
10 HOME
20 FOR L=1 TO 250:NEXT L
30 POKE 1063,47
40 FOR L=1 TO 250:NEXT L
50 GOTO 10
```

92 Screen Full of Garbage

Fill your video screen with @ square dots, starting from the upper left, moving down and across to the lower right.

Push the RESET key to stop the action and regain control from the looping line 160.

Program Listing

```
100 HOME
110 FOR Y=1 TO 39
120 FOR X=1 TO 47
130 PLOT Y,X
140 NEXT X
150 NEXT Y
160 GOTO 160
```

93 Create A Table

This program generates a table of values, as a

demonstration on how to set up a table on the video display.

Subroutine lines 900 and 910 generate random numbers in the range of zero to 99. Lines 20 and 30 find how many times through the random number generator it takes to get a number greater than 50. The answer is stored in A.

Lines 40 and 50 do it again and store the answer in B. Lines 60 and 70 do it and store in C.

Line 10 prints the table heading and line 100 displays the results. Line 110 causes the whole operation to repeat until you have a table of 20 lines on the screen.

Program Listing

```
  5 HOME
 10 PRINT "A","B","C":PRINT
 20 GOSUB 900
 30 IF X>50 THEN A=A+1:GOTO 20
 40 GOSUB 900
 50 IF X>50 THEN B=B+1:GOTO 20
 60 GOSUB 900
 70 IF X>50 THEN C=C+1:GOTO 20
100 PRINT A,B,C:IF T=19 THEN END
110 A=Ø:B=Ø:C=Ø:T=T+1:GOTO 20
120 END
900 X=INT(100*(RND(1)))
910 RETURN
```

94 Flashing Character Screen

We use ASCII number 100 to quickly fill the video screen with dollar signs. They flash continuously.

You may change the number 100 in line 30 to fill the screen with some other character.

Use this program as an attention getter in a game or some other program. If you tie more program on the end, after line 40, use HOME in line 50 to clear the screen.

Program Listing

```
10 HOME
20 FOR L=1024 TO 2039
30 POKE L,100
40 NEXT L
```

95 Organizing Results

This program computes hourly wages for an office bookkeeper. We include it here as an example of how to design a nice output on your video screen.

Your APPLE is a powerful tool and you should present results of its computations in the most eye-pleasing manner.

On the input side, this program includes good title box and three brief questions. On the output side, it presents a well-organized chart.

Program Listing

```
 10 HOME:PRINT CHR$(7)
 20 PRINT:PRINT
 30 PRINT"****************"
 40 PRINT"* HOURLY WAGES *"
 50 PRINT"****************"
 60 PRINT:PRINT
100 INPUT"EMPLOYEE'S NAME:  ";N
110 PRINT CHR$(7)
120 INPUT"HOURLY PAY RATE:  ";P
130 PRINT CHR$(7)
140 INPUT"TOTAL NUMBER OF HOURS WORKED
    :  ";H
150 PRINT CHR$(7)
200 IF H=40 OR H<40 THEN RH=H:OH=Ø
    :GOTO 300
210 IF H>40 THEN OH=H-40:RH=40
300 OP=1.5*P:PA=RH*P:PB=OH*OP
310 PY=PA+PB
```

```
320 PY=INT(100*PY+.5)/100
400 HOME
410 PRINT"EMPLOYEE:",N$
420 PRINT
430 PRINT"TOTAL HOURS:",H
440 PRINT
450 PRINT"REGULAR HOURS:",RH
460 PRINT"REGULAR RATE:","$";P
470 PRINT"TOTAL REGULAR:","$";PA
480 PRINT
490 PRINT"OVERTIME HOURS:",OH
500 PRINT"OVERTIME RATE:","$";OP
510 PRINT"TOTAL OVERTIME:","$";PB
520 PRINT
530 PRINT"GROSS PAY:","$";PY
600 FOR L=1 TO 6:PRINT:NEXT L
610 PRINT"TO COMPUTE WAGES FOR A"
620 PRINT"DIFFERENT EMPLOYEE, PRESS ANY KEY"
630 GET KY$
640 GOTO 10
```

Sample Run

```
****************
* HOURLY WAGES *
****************

EMPLOYEE'S NAME:
JONES       RETURN
HOURLY PAY RATE:
5.00        RETURN
TOTAL NUMBER OF HOURS WORKED:
50          RETURN


EMPLOYEE:  JONES

TOTAL HOURS:     50

REGULAR HOURS:   40
REGULAR RATE:    $5
TOTAL REGULAR:   $200
```

```
OVERTIME HOURS: 10
OVERTIME RATE:  $7.5
TOTAL OVERTIME: $75

GROSS PAY:      $2.75


TO COMPUTE WAGES FOR A
DIFFERENT EMPLOYEE, PRESS ANY KEY
```

96 Centered Message

Think up a message of up to 34 characters and spaces. Type it into your APPLE in response to the request here at line 110.

The computer will highlight your message by centering it on the video screen.

Use this handy centering program for titles and other important parts of longer programs. It makes a good subroutine.

Program Listing

```
100 HOME:CLEAR
110 INPUT"NEW MESSAGE TO BE CENTERED: ";M$
120 LM=LEN(M$)
130 LT=LM+4
140 FOR L=1 TO LT
150 AS$=AS$+"*"
160 NEXT L
170 P=(40-LT)/2
180 HOME
190 PRINT TAB(P) AS$
200 PRINT TAB(P) "* ";M$;" *"
210 PRINT TAB(P) AS$
220 FOR L=1 TO 9:PRINT:NEXT L
230 CLEAR:GOTO 110
```

97 Manual Box Mover

The computer can be made to take your message (of up to 34 characters and spaces) and center that message, and draw a box around it to highlight it.

Line 10 asks for a word or words of up to 34 total characters and spaces. Lines 30-80 draw a box around it. Line 100 positions it at the bottom of the video screen. Lines 110 to 130 display it, in its box.

At this point the computer waits for you to press any key (line 200). When you press a key on the keyboard, the program progresses to line 210. The entire message box is pushed *up* one line.

Each key press pushes the box up one more line. Imagination!

Program Listing

```
 10 HOME
 20 INPUT"MESSAGE TO BE BOXED ";M$
 30 LM=LEN(M$)
 40 LT=LM+4
 50 FOR L=1 TO LT
 60 AS$=AS$+"*"
 70 NEXT L
 80 P=(40-LT)/2
 90 HOME
100 FOR L=1 TO 20:PRINT:NEXT L
110 PRINT TAB(P) AS$
120 PRINT TAB(P) "* ";M$;" *"
200 GET KY$
210 CALL -912
220 GOTO 200
```

98 Repeating Box Move

Did you like the manual box mover? How about an automatic upward flight?

You give the computer the message (up to 34 spaces and characters) and it boxes it, centers, and locates it at the bottom of the screen.

It automatically moves up toward the top of the screen. As it leaves the top of the screen, it starts over at the bottom, climbing again to the top. And on and on.

The upward scrolling speed is controlled by the time-delay loop in line 220. To slow things down, increase the number 100 in line 220. To speed up the scroll upward, decrease the number 100 in line 220.

Program Listing

```
 10 HOME
 20 INPUT "MESSAGE TO REPEAT";M$
 30 LM=LEN(M$)
 40 LT=LM+4
 50 FOR L=1 TO LT
 60 XQ$=XQ$+"*"
 70 NEXT L
 80 P=(40-LT)/2
 90 HOME
100 FOR L=1 TO 20:PRINT:NEXT L
110 PRINT TAB(P) XQ$
120 PRINT TAB(P) "* ";M$;" *"
130 PRINT TAB(P) XQ$
200 FOR L=1 TO 19
210 CALL -912
220 FOR LL=1 TO 100:NEXT LL
230 NEXT L
240 GOTO 90
```

99 Flashing Screen Block

Here we have two versions of a one-line program. Either works alone. Both create large flashing blocks of characters on the APPLE video display.

The first program covers the entire screen. The second program only covers the top third of the screen.

First Program:

```
10 FOR L=1 TO 240
   :A$=A$+"*"
   :NEXT L
   :PRINT A$
   :CLEAR
   :GOTO 20
```

Second Program:

```
10 FOR L=1 TO 240
   :A$=A$+"*"
   :NEXT L
   :HOME
   :PRINT A$
   :CLEAR
   :GOTO 20
```

Odds & Ends

100 Through A Looking Glass

Why? We'll leave that up to you. But it is important to know this can be done.

LIST is in line 210. Running LIST within a program does not cause operations to cease as in some computers. Once again the APPLE is more versatile!

Try it. It's unusual.

Program Listing

```
 10 HOME
 20 PRINT "TO RUN,PRESS R"
 30 INPUT "TO LIST, PRESS L";A$
 40 IF A$="R" THEN GOTO 100
 50 IF A$="L" THEN GOTO 200
 60 GOTO 20
100 PRINT
110 INPUT "TWO NUMBERS ARE ";X,Y
120 PRINT "FIRST DIVIDED BY SECOND = "
    ;X/Y
130 PRINT
140 GOTO 20
200 PRINT
210 LIST
220 PRINT
230 GOTO 20
```

101 List Breaker

Sometimes it's convenient to place several programs at higher line numbers for "storage." Lower line numbers then are free for new uses.

For ease in sorting out the programs later, we insert a breaker **between** the programs. Use REM and 25 asterisks on each of two lines as we show in our sample program here. See lines 5470 and 5480.

Later, when you LIST and are scanning down the lines, you can spot your breaker lines of asterisks.

By the way, these program lines 5440-5510 will not run as a stand-alone program. They are for display purposes only.

Program Listing

```
5440 P=5:Q=55:X=11:Y=22
5450 PRINT Q/P
5460 PRINT Y/X
5470 REM ************************
5480 REM ************************
5490 HOME
5500 FOR L=1 TO 1000
5510 PRINT L
5520 NEXT L
```

102 Memory Review

This useful program examines a range of memory locations you specify and reports what it finds there. It displays the memory location number alongside the contents, for convenience.

Program Listing

```
 10 HOME
 20 PRINT TAB(5)"MEMORY REVIEW"
 30 PRINT:PRINT"SEARCH FOR CONTENTS FROM"
 40 INPUT"LOW MEMORY LOCATION NUMBER
    : ";LM
 50 INPUT"TO HIGH MEMORY LOCATION NUMBER
    : ";HM
 60 PRINT:PRINT"FAST SPEED","PRESS F"
 70 PRINT"SLOW SPEED","PRESS S"
 80 GET KY$
 90 IF KY$="F" THEN SP=50:GOTO 120
100 IF KY$="S" THEN SP=500:GOTO 120
110 GOTO 80
120 HOME:PRINT"MEMORY","CONTENT","VALUE"
```

```
130 FOR L=LM TO HM
140 FOR T=1 TO SP:NEXT T
150 PK=PEEK(L)
160 PRINT L,PK,CHR$(PK)
170 NEXT L
200 FOR L=1 TO 5:PRINT:NEXT L
210 PRINT"END OF RUN"
220 PRINT"FOR MORE, PRESS ANY KEY"
230 GET KY$
240 CLEAR:GOTO 10
```

103 Wipeout!

Warning: handle with care!

Careless operation of this program can cause you a lot of extra work.

Key in the program. Run it. When it asks for the password, be sure to give it what it wants or it will erase itself. That's right, the entire contents of program memory down the tubes!

Here's how it works:

The password, in this case Tracey , is asked by line 20. You give it a password. In line 30, your answer is compared to the true password. If correct, action goes to line 50. If incorrect, the program goes to line 40.

The NEW statement in line 40 erases everything from program memory.

You can change the password in line 30 to any letters, numbers or keyboard symbols of your choice. Watch out when testing. A wrong password can cause a lot of retyping.

Program Listing

```
10 HOME
20 INPUT "WHAT IS THE PASSWORD ?";P$
30 IF P$="TRACEY" THEN GOTO 50
40 NEW
50 PRINT "CORRECT"
60 PRINT "THE PASSWORD IS ";P$
```

Appendix A:
The APPLE's BASIC Words

Here is a handy list of all the words in Applesoft BASIC. The computer uses only one byte of memory space to store each full word of these instructions. That compares with one byte per character for other information you type into your APPLE. For example, the instruction RESTORE is stored all in one byte of memory while someone's name, Jones, for instance, would take up five bytes of memory space.

&

ABS
AND
ASC
AT
ATN

CALL
CHR$
CLEAR
COLOR=
CONT
COS

DATA
DEF
DEL
DIM
DRAW

END
EXP

FLASH
FN
FOR
FRE

GET
GOSUB
GOTO
GR

HCOLOR=
HGR

HGR2
HIMEM:
HLIN
HOME
HPLOT
HTAB

IF
IN#
INPUT
INT
INVERSE

LEFT$
LEN
LET
LIST
LOAD
LOG
LOMEM:

MID$

NEW
NEXT
NORMAL
NOT
NOTRACE

ON
ONERR
OR

PDL
PEEK
PLOT
POKE
POP
POS
PRINT
PR#

READ
RECALL
REM
RESTORE
RESUME

RETURN
RIGHT$
RND
ROT=
RUN

SAVE
SCALE=
SCRN(
SGN
SHLOAD
SIN
SPC(
SPEED=
SQR
STEP
STOP
STORE
STR$

TAB(
TAN
TEXT
THEN
TO
TRACE

USR

VAL
VLIN
VTAB

WAIT

XPLOT
XDRAW

The ampersand and XPLOT are not useful, generally. COLOR, HCOLOR, SCALE, SPEED and ROT must be followed immediately by an equal sign. SCRN, SPC and TAB must be followed immediately by a left parenthesis. HIMEM and LOMEM must be followed immediately by a colon. ATN must have no space between its letters. TO must not be preceded by the letter A or the computer will think you want AT.

Appendix B: Error Messages

Your APPLE will tell you what went wrong when you make a programming mistake. Here are the messages you might receive:

?CAN'T CONTINUE ERROR

You are incorrectly trying to continue running a program after an error has occurred, or after a line has been deleted or added to a program. Or the program does not exist.

?DIVISION BY ZERO ERROR

You can't divide a number by zero as you are attempting.

?FORMULA TOO COMPLEX ERROR

You are trying to use more than two IF/THEN instructions at once.

?ILLEGAL DIRECT ERROR

When you try to use your APPLE like a calculator, you are using it in the *immediate execution* mode. When you try to run a previously-entered program, you use the *deferred execution* mode. You cannot use the instructions INPUT, DEF FN, GET or DATA in the *immediate* mode. This error message says you are incorrectly trying to do that.

?ILLEGAL QUANTITY ERROR

You are trying to use a number beyond the range of ability of your APPLE.

?NEXT WITHOUT FOR ERROR

Your NEXT has no matching FOR.

?OUT OF DATA ERROR

You are trying to READ but all DATA has been used.

?OUT OF MEMORY ERROR

Your program is too large. Or you have used too many variables. Or your loops or subroutines nesting is too complex for the APPLE to handle.

?OVERFLOW ERROR

The result of a computation is just too large for the APPLE.

?REDIM'D ARRAY ERROR

You are trying incorrectly to redimension an array you already have dimensioned.

?RETURN WITHOUT GOSUB ERROR

You are trying to use the RETURN instruction but you didn't GOSUB in the first place. RETURN must always be associated with a GOSUB instruction.

?STRING TOO LONG ERROR

You are trying to create a string of more than 255 characters.

?BAD SUBSCRIPT ERROR

You are trying to use an array bigger than the dimension you previously set.

?SYNTAX ERROR

You are not using BASIC correctly. Your punctuation or letters or symbols are wrong.

?TYPE MISMATCH ERROR

You're trying to match apples with oranges. You are trying to mix numeric and string variables. The computer is looking for a numeric value and you are giving it a string or else it wants a string and you are giving it a numeric value.

?UNDEF'D STATEMENT ERROR

You are trying to jump to a nonexistent line. Check your GOTO, GOSUB or THEN instruction.

?UNDEF'D FUNCTION ERROR

You are trying to use a "user defined" function but you haven't previously defined such a function.

books from ARCsoft Publishers

For the TRS-80 PC-1, PC-2, Sharp PC-1211, PC-1500:

99 Tips & Tricks for the New Pocket Computers (PC-2/PC-1500)
128 pages $7.95 ISBN 0-86668-019-5

101 Pocket Computer Programming Tips & Tricks
128 pages $7.95 ISBN 0-86668-004-7

Pocket Computer Programming Made Easy
128 pages $8.95 ISBN 0-86668-009-8

Murder In The Mansion and Other Computer Adventures
96 pages $6.95 ISBN-0-86668-501-4

50 Programs in BASIC for Home, School & Office
96 pages $9.95 ISBN 0-86668-502-2

50 MORE Programs in BASIC for Home, School & Office
96 pages $9.95 ISBN 0-86668-003-9

Pocket-BASIC Coding Form programming worksheet tablets
40-sheet pad $2.95 ISBN 0-86668-801-3

For the TRS-80 Color Computer:

101 Color Computer Programming Tips & Tricks
128 pages $7.95 ISBN 0-86668-007-1

55 Color Computer Programs for Home, School & Office
128 pages $9.95 ISBN 0-86668-005-5

55 MORE Color Computer Programs for Home, School & Office
112 pages $9.95 ISBN 0-86668-008-X

Color Computer Graphics
128 pages $9.95 ISBN 0-86668-012-8

The Color Computer Songbook
96 pages $7.95 ISBN 0-86668-011-X

My Buttons Are Blue and Other Love Poems
96 pages $4.95 ISBN 0-86668-013-6

Color Computer BASIC Coding Form program worksheet tablets
40-sheet pad $2.95 ISBN 0-86668-802-1

For the APPLE Computer:

101 APPLE Computer Programming Tips & Tricks
128 pages $8.95 ISBN 0-86668-015-2

33 New APPLE Computer Programs for Home, School & Office
96 pages $8.95 ISBN 0-86668-016-0

APPLE Computer BASIC Coding Form program worksheet tablets
40-sheet pad $2.95 ISBN 0-86668-803-X

For the ATARI 400 and 800 computers:

101 ATARI Computer Programming Tips & Tricks
128 pages $8.95 ISBN 0-86668-022-5

31 New ATARI Computer Programs for Home, School & Office
96 pages $8.95 ISBN 0-86668-018-7

ATARI Computer BASIC Coding Form program worksheet tablets
40-sheet pad $2.95 ISBN 0-86668-806-4

books from ARCsoft Publishers

For the Sinclair ZX-81 and TIMEX/Sinclair 1000 computers:

101 TIMEX 1000 and Sinclair ZX-81 Programming Tips & Tricks
128 pages $7.95 ISBN 0-86668-020-9

37 TIMEX 1000 & Sinclair ZX-81 Programs for Home, School & Office
96 pages $8.95 ISBN 0-86668-021-7

TIMEX 1000 & Sinclair ZX-81 BASIC Coding Form program worksheets
40-sheet pad $2.95 ISBN 0-86668-807-2

For the electronics hobbyist:

25 Quick-N-Easy Electronics Projects
96 pages $4.95 ISBN 0-86668-023-3

25 Electronics Projects for Beginners
96 pages $4.95 ISBN 0-86668-017-9

25 Easy-To-Build One-Night & Weekend Electronics Projects
96 pages $4.95 ISBN 0-86668-010-1

Computer program-writing worksheets:
Each in tablet of 40 sheets with stiff backing

Universal BASIC Coding Form	$2.95
IBM Personal Computer BASIC Coding Form	$2.95
ATARI Computer BASIC Coding Form	$2.95
Pocket Computer BASIC Coding Form	$2.95
APPLE Computer BASIC Coding Form	$2.95
Color Computer BASIC Coding Form	$2.95
TIMEX 1000/Sinclair ZX-81 BASIC Form	$2.95

ISBN: International Standard Book Number

ARCsoft Publishers

Post Office Box 132
Woodsboro, Maryland 21798

APPLE Computer Book I

101 APPLE Computer Programming Tips & Tricks

by Fred White

Here's a mammoth collection of practical, useful, efficient programming techniques and operating shortcuts right out of a master programmer's notebook. Loaded with hints, secrets, shortcuts, tips, tricks and easy-to-follow instructions, this book tells you how to handle routine programming chores more quickly, do special effects, make your computer work for you — faster and more efficiently.

Each of the 101 APPLE Computer programming tips and tricks in this book features a complete program which will run all by itself, or as part of a larger set of instructions. All 101 programs are tested and ready to type in and run.

Learn insider's how-to secrets for using the exciting BASIC words, HOME, PEEK, CALL, INPUT, LEN, GOSUB, RETURN, READ, DATA, RESTORE, VAL, PRINT, DIM, LEFT$, CHR$, FOR, NEXT, TAB, POKE, HLIN, PLOT, SOUND, and all the rest.

Sections in this book include *fun & games, text on text, gee whiz, number crunching, business matters, graphics* and *odds & ends.* Special techniques are here for graphics; sound from your APPLE; colorful program title and billboards; sound cues; poking and peeking into memory; searching, ordering, sorting and error trapping; making a custom cursor; secret messages; writing and playing games; batting averages; math flasher; ASCII character review; percentages; dollars and cents; decision maker; list breaker; random numbers; lookup table; killing time; three-digit mystery; news ticker; invoicing; passwords; and many more.

You'll learn what to do, how to do it, when to make changes and when not to, all in this extensive info-packed APPLE computer programmer's handbook of tips and tricks.

ARCsoft Publishers
WOODSBORO, MARYLAND

ISBN 0-86668-015-2