A COMPUTE! Books Publication

$12.95

33 PROGRAMS FOR THE TI-99/4A

Brian Flynn

Challenging games, money management, data sorting, financial forecasting, and many other exciting programs and applications for your TI-99/4A with Extended BASIC.

Brian Flynn

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies
Greensboro, North Carolina

Printed in the United States of America

ISBN 0-942386-42-6

10 9 8 7 6 5 4 3 2 1

COMPUTE! Publications, Inc., Post Office Box 5406, Greensboro, NC 27403, (919) 275-9809, is one of the ABC Publishing Companies and is not associated with any manufacturer of personal computers. TI-99/4A is a trademark of Texas Instruments.

Contents

Foreword

As you've undoubtedly discovered, your Texas Instruments computer is a versatile machine. Along with TI's Extended BASIC, it can give you computing capabilities once found only on more complex machines—and *33 Programs for the TI-99/4A* will help you put that computing power to work.

This book covers a variety of applications and is sure to offer something for almost everybody. Money management programs can have an immediate impact on your personal finances. Statistical and numerical routines add sophisticated analytical tools to your software library. And, of course, there are games that will give you hours of challenging fun.

Like other COMPUTE! Books, *33 Programs for the TI-99/4A* contains complete program listings for you to type in and run. The programs are easy to use, and the accompanying descriptions and explanations are clear and informative. All 33 programs require TI's Extended BASIC cartridge; they can be run as is or easily customized to suit your own particular needs.

If you've seen COMPUTE!'s other books for the TI-99/4A computer, you've already tapped the power that lies beneath your keyboard. With these utilities, applications, and games, you have an even more versatile and powerful computer at your command.

Chapter 1

Money Management

Money Management

Everybody worries about money—and with soaring inflation slashing purchasing power one year while depression-level unemployment saps income the next, those nerve-racking worries aren't likely to end. But with the programs presented here, your TI computer can ease the strain.

- "Effective Yield on an Investment" computes the actual return on your money, at different interest rates and on different compounding schedules.
- "Treasury Bill Yields" uses official U.S. Treasury formulas to compute discount rates and yields on three-month, six-month, and one-year T bills.
- "IRA Planner" evaluates your Individual Retirement Account, both in current dollars and in dollars adjusted for inflation, at maturity.
- "Municipal Bond Buyers" computes the cost of tax-free municipal bonds, their tax-free and taxable-equivalent yields, and their future worth.
- "Loan Payment" computes both the monthly payment on a loan and the total payment made over the life of the loan.
- "Mortgage Payment" computes yearly, quarterly, or monthly mortgage payments, the total payment over the life of the loan, and the amount paid to principal and to interest.

Effective Yield on an Investment

"Money has value over time." This is an established maxim in economics. You prove it every time you put money in your savings account. You give the teller $500, for example, only on condition that you get back more than $500 in the future. You want to earn *interest* on your savings.

Interest may be compounded yearly, twice yearly, quarterly, monthly, daily, or continuously. But when interest is compounded more often than once a year, nominal and effective interest rates differ. For example, a $500 investment, with interest compounded twice yearly at a $10 *nominal* rate, yields $500 x $(1 + 0.10/2)^2$ in a year, or $551.25. Thus, the *effective* interest rate, or the actual percentage return on your money, is 10.25 percent per annum.

With the help of the Effective Yield program, your TI computer can calculate a host of effective interest rates for any

nominal rate that you choose. Simply enter the amount of the principal, and the display will show you the dollar impact of various frequencies of compounding. For example, $10,000 invested at a 12 percent nominal rate, compounded annually, produces $11,200 in a year. Continuous compounding, on the other hand, yields $11,275, or an additional $75.

Figure 1-1. Effective Yield on an Investment

NOMINAL YIELD = 12%

FREQUENCY OF COMPOUNDING	EFFECTIVE YIELD
ANNUAL	12.000%
TWICE YEARLY	12.360
QUARTERLY	12.551
MONTHLY	12.683
DAILY	12.747
CONTINUOUSLY	12.750

Program 1-1. Effective Yield on an Investment

```
100 REM EFFECTIVE YIELD ON AN INVESTMENT
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER NOMINAL YIELD
160 GOSUB 320
170 REM COMPUTE EFFECTIVE YIELD
180 GOSUB 560
190 REM DISPLAY RESULTS
200 GOSUB 660
210 END
220 REM INITIALIZE
230 CALL CLEAR
240 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES"
250 DISPLAY AT(2,1):"EFFECTIVE YIELDS ON AN"
260 DISPLAY AT(3,1):"INVESTMENT."
270 DISPLAY AT(5,1):"IT DOES THIS FOR VARIOUS"
280 DISPLAY AT(6,1):"FREQUENCIES OF INTEREST" :: D
    ISPLAY AT(7,1):"COMPOUNDING."
290 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
300 ACCEPT AT(24,25):Z$
310 RETURN
320 REM NOMINAL YIELD
330 REM FREQUENCIES
340 GOSUB 380
```

```
350 REM ENTRY
360 GOSUB 440
370 RETURN
380 REM FREQUENCIES
390 DATA ANNUAL,TWICE YEARLY,QUARTERLY,MONTHLY,DAI
    LY,CONTINUOUSLY
400 FOR I=1 TO 6
410 READ FREQ$(I)
420 NEXT I
430 RETURN
440 REM ENTRY
450 CALL CLEAR
460 DISPLAY AT(1,1):"PLEASE ENTER THE NOMINAL"
470 DISPLAY AT(2,1):"YIELD OF YOUR INVESTMENT."
480 DISPLAY AT(4,1):"FOR EXAMPLE, IF THE YIELD"
490 DISPLAY AT(5,1):"IS 7% A YEAR, ENTER 7."
500 DISPLAY AT(7,1):"IF THE YIELD IS 12% A YEAR,"
510 DISPLAY AT(8,1):"ENTER 12, AND SO ON."
520 DISPLAY AT(11,1):"YIELD = ? "
530 ACCEPT AT(11,11)BEEP VALIDATE(NUMERIC,""):NOMY
    D$
540 IF NOMYD$="" THEN 530 ELSE NOMYD=VAL(NOMYD$)
550 RETURN
560 REM COMPUTE
570 NOMYD=NOMYD/100
580 DEF YD(F)=(((1+NOMYD/F)^F)-1)*100
590 YIELD(1)=YD(1)
600 YIELD(2)=YD(2)
610 YIELD(3)=YD(4)
620 YIELD(4)=YD(12)
630 YIELD(5)=YD(365)
640 YIELD(6)=(EXP(NOMYD)-1)*100
650 RETURN
660 REM DISPLAY RESULTS
670 CALL CLEAR
680 CALL HCHAR(1,3,61,28)
690 DISPLAY AT(2,5):"NOMINAL YIELD =";NOMYD*100;"%
    "
700 CALL HCHAR(3,3,61,28)
710 DISPLAY AT(5,3):"FREQUENCY"
720 DISPLAY AT(6,7):"OF";TAB(18);"EFFECTIVE"
730 DISPLAY AT(7,2):"COMPOUNDING";TAB(20);"YIELD"
740 IMAGE ####.### %
750 ROW=10
760 FOR I=1 TO 6
770 DISPLAY AT(ROW,3):FREQ$(I)
780 DISPLAY AT(ROW,17):USING 740:YIELD(I)
790 ROW=ROW+2
```

```
800 NEXT I
810 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
820 CALL HCHAR(23,3,61,28)
830 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
840 ACCEPT AT(24,26):Z$
850 RETURN
```

Treasury Bill Yields

The federal budget is in deficit when the government spends more than it takes in, and the Treasury tries to help make up the difference by selling bills and bonds. A Treasury bill is a *promise to pay*. You give the government a certain amount of money, and it promises to pay back the principal plus interest at a later date.

Treasury bills (T-bills) are purchased from the Bureau of Public Debt, Federal Reserve Banks, some commercial banks, and brokerage houses. T-bills come in three varieties: three-month, six-month, and one-year. They sell on a discount basis. This means that we receive our interest payment up front, with our initial investment or principal returned when the bill comes due. T-bills offer competitive yields and come in denominations of $10,000 or more, in multiples of $5000.

This TI program calculates the discount rate and yield on a T-bill purchase using official U.S. Treasury formulas. It first asks you what term—three, six, or twelve months—your bill covers. You'll also have to enter the discount rate (that is, the interest rate) and the denomination. Finally, you'll need to tell the computer whether or not it's a leap year.

Figure 1-2. Treasury Bill Yields

26 WEEK TREASURY BILL

TOTAL	=	$10000.00
INVESTMENT	=	$ 9400.00
INTEREST	=	$ 600.00
DISCOUNT RATE	=	11.87%
YIELD	=	12.80%

For instance, suppose you invest $10,000 in a 26-week bill. You receive $600 in interest right away, with the principal returned in half a year. The discount rate in this case is the percentage return on the $10,000, or 11.87 percent annually.

Entering these figures and running the program yields the output shown in Figure 1-2. As you can see, since you get the $600 almost immediately, your actual investment is only $9400. The true yield, or the return on the $9400, is 12.80 percent per annum.

One word of warning. In computing discount rates, the Treasury uses a banker's year of 360 days instead of 365, as the technical note explains. Further, interest is not compounded in computing three- and six-month yields but is simply summed. If you find this strange, write the Treasury!

Notes on the Program

1. Discount rates are tallied using the formula:

$$\frac{\text{Interest Payment}}{\$10{,}000} \times \frac{360}{D}$$

where D is the number of days until the bill matures. D will equal 91 for a three-month bill, 182 for a six-month bill, and 364 for a one-year bill.

In this example, the interest payment is $600 and the term of the bill is six months. Therefore, the discount rate is:

$600/($10,000 x 360/182) = 0.1187, or 11.87% per year

2. Yields for three- and six-month bills are tallied as follows:

$$\frac{\text{Interest Payment}}{\$10{,}000 - \text{Interest}} \times \frac{Y}{D}$$

where D is the number of days until maturity and Y is the number of days in the year, 365 or 366. In our example, the yield is:

$600/($10,000 − $600) × 365/182 = 0.1280, or 12.80% per year

The yield on a one-year bill is computed using a very elaborate formula that compounds interest semiannually.

Program 1-2. Treasury Bill Yields

```
100 REM T-BILL YIELDS
130 REM HEADING
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 370
```

```
170 REM COMPUTE
180 GOSUB 870
190 REM DISPLAY RESULTS
200 GOSUB 1250
210 END
220 REM HEADING
230 CALL CLEAR
240 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
250 DISPLAY AT(2,1):"YIELD ON A U.S. TREASURY"
260 DISPLAY AT(3,1):"BILL, OR A T-BILL FOR SHORT."
270 DISPLAY AT(5,1):"A T-BILL IS A 'PROMISE TO"
280 DISPLAY AT(6,1):"PAY' SOLD BY THE TREASURY."
290 DISPLAY AT(8,1):"YOU GIVE THE TREASURY $10K,"
300 DISPLAY AT(9,1):"FOR EXAMPLE, AND IT GIVES"
310 DISPLAY AT(10,1):"YOU AN INTEREST PAYMENT" ::
    DISPLAY AT(11,1):"RIGHT AWAY."
320 DISPLAY AT(13,1):"YOU GET THE $10K BACK WHEN"
330 DISPLAY AT(14,1):"THE BILL COMES DUE."
340 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
350 ACCEPT AT(24,25):Z$
360 RETURN
370 REM ENTER DATA
380 REM DURATION
390 GOSUB 450
400 REM INTEREST PAYMENT
410 GOSUB 580
420 REM SIZE OF INVESTMENT
430 GOSUB 720
440 RETURN
450 REM DURATION
460 CALL CLEAR
470 DISPLAY AT(1,1):"T-BILLS MATURE OR COME DUE"
480 DISPLAY AT(2,1):"IN THE FOLLOWING LENGTHS OF"
490 DISPLAY AT(3,1):"TIME:"
500 DISPLAY AT(5,5):"1. 13 WEEKS"
510 DISPLAY AT(6,5):"2. 26 WEEKS"
520 DISPLAY AT(7,5):"3. 52 WEEKS"
530 DISPLAY AT(10,1):"WHICH TYPE OF T-BILL WOULD"
540 DISPLAY AT(11,1):"YOU LIKE ? "
550 ACCEPT AT(11,12)BEEP VALIDATE("123","")SIZE(1)
    :T$
560 IF T$="" THEN 550 ELSE TYPE=VAL(T$)
570 RETURN
580 REM INTEREST PAYMENT
590 CALL CLEAR
600 DISPLAY AT(1,1):"FOR EACH $10K INVESTED,"
610 DISPLAY AT(2,1):"PLEASE TELL ME HOW MUCH"
620 DISPLAY AT(3,1):"INTEREST YOU EITHER RECEIVED"
630 DISPLAY AT(4,1):"BACK FROM THE TREASURY OR"
```

```
640 DISPLAY AT(5,1):"EXPECT TO RECEIVE BACK."
650 DISPLAY AT(8,1):"DISCOUNT"
660 DISPLAY AT(9,1):"PAYMENT = ? "
670 ACCEPT AT(9,12)BEEP VALIDATE(NUMERIC,""):P$
680 IF P$="" THEN 670 ELSE PAY=VAL(P$)
690 IF PAY<=0 THEN DISPLAY AT(23,1):"I KNOW THE GO
    VERNMENT IS" :: DISPLAY AT(24,1):"CHINTZY, BUT
    THAT'S CRAZY!" :: GOTO 670
700 IF PAY>=10000 THEN DISPLAY AT(23,1):"COME ON!
    THAT'S WISHFUL" :: DISPLAY AT(24,1):"THINKING!
    " :: GOTO 670
710 RETURN
720 REM SIZE OF INVESTMENT
730 CALL CLEAR
740 DISPLAY AT(1,1):"T-BILLS COME IN DENOMINA-"
750 DISPLAY AT(2,1):"TIONS OF 10K OR MORE, IN"
760 DISPLAY AT(3,1):"MULTIPLES OF 5K."
770 DISPLAY AT(5,1):"PLEASE ENTER THE SIZE OF"
780 DISPLAY AT(6,1):"YOUR INVESTMENT."
790 DISPLAY AT(8,1):"THAT IS, 10 FOR $10K, 15 FOR"
800 DISPLAY AT(9,1):"$15K, AND SO ON."
810 DISPLAY AT(11,1):"SIZE = ?"
820 ACCEPT AT(11,10)BEEP VALIDATE(DIGIT,""):S$
830 IF S$="" THEN 820 ELSE SZE=1000*VAL(S$)
840 SZE=(INT(SZE/5000))*5000
850 IF SZE<10000 THEN DISPLAY AT(23,1):"SORRY, AT
    LEAST $10K NEEDED" :: GOTO 820
860 RETURN
870 REM COMPUTE
880 REM CHECK FOR LEAP YEAR
890 GOSUB 1160
900 IF LEAP$="Y" THEN DAYS=366 ELSE DAYS=365
910 REM NUMBER OF DAYS UNTIL BILL MATURES
920 IF TYPE=1 THEN N=91
930 IF TYPE=2 THEN N=182
940 IF TYPE=3 THEN N=364
950 REM DISCOUNT RATE
960 RATE=(PAY/10000)*(360/N)
970 REM YIELD
980 PRINCIPAL=10000-PAY
990 ON TYPE GOSUB 1060,1060,1090
1000 REM COMPOSITION OF INVESTMENT
1010 REM NUMBER OF 10K INCREMENTS
1020 INCR=SZE/10000
1030 REM TOTAL INTEREST
1040 INTEREST=INCR*PAY
1050 RETURN
```

```
1060 REM 3 &  6 MONTH BILLS
1070 YIELD=(PAY/PRINCIPAL)*(DAYS/N)
1080 RETURN
1090 REM 1 YEAR BILLS
1100 REM USE QUADRATIC FORMULA
1110 A=364/(2*DAYS)-.25
1120 B=364/DAYS
1130 C=-PAY/PRINCIPAL
1140 YIELD=(-B+SQR(B*B-4*A*C))/(2*A)
1150 RETURN
1160 REM CHECK FOR LEAP YEAR
1170 CALL CLEAR :: DISPLAY AT(1,1):"IN ORDER TO CO
     MPUTE THE"
1180 DISPLAY AT(2,1):"YIELD ON YOUR INVESTMENT,"
1190 DISPLAY AT(3,1):"I NEED TO KNOW IF THIS IS"
1200 DISPLAY AT(4,1):"A LEAP YEAR."
1210 DISPLAY AT(7,1):"IS IT (Y/N) ?"
1220 ACCEPT AT(7,15)BEEP VALIDATE("YN","")SIZE(1):
     LEAP$
1230 IF LEAP$="" THEN 1220
1240 RETURN
1250 REM RESULTS
1260 CALL CLEAR
1270 CALL HCHAR(1,3,61,28)
1280 IF TYPE=1 THEN S$="13 "
1290 IF TYPE=2 THEN S$="26 "
1300 IF TYPE=3 THEN S$="52 "
1310 DISPLAY AT(2,4):S$;"WEEK TREASURY BILL"
1320 CALL HCHAR(3,3,61,28)
1330 IMAGE = $##########.##
1340 DISPLAY AT(5,1):"TOTAL" :: DISPLAY AT(6,1):"I
     NVESTMENT" :: DISPLAY AT(6,12):USING 1330:SZE
1350 DISPLAY AT(8,1):"INTEREST" :: DISPLAY AT(8,12
     ):USING 1330:INTEREST
1360 DISPLAY AT(10,1):"PRINCIPAL" :: DISPLAY AT(10
     ,12):USING 1330:SZE-INTEREST
1370 IMAGF = ####.## %
1380 DISPLAY AT(15,1):"DISCOUNT RATE"
1390 DISPLAY AT(15,15):USING 1370:RATE*100
1400 DISPLAY AT(16,1):"YIELD"
1410 DISPLAY AT(16,15):USING 1370:YIELD*100
1420 CALL HCHAR(23,3,61,28)
1430 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1440 ACCEPT AT(24,26):Z$
1450 RETURN
```

IRA Planner

An IRA is an Individual Retirement Account. People are creating these accounts in record numbers, primarily for tax reasons.

IRAs offer a number of advantages. Any earnings on an IRA are tax-free until you start withdrawing funds. In addition, up to $2000 (for single persons) can be added to an IRA each year while simultaneously being subtracted from gross taxable income. That means that anyone in a 50 percent tax bracket can actually reap IRA benefits for half the price.

There is a catch, however. You can't withdraw money from an IRA before age 59-1/2 without suffering a stiff tax penalty, and you must begin to make withdrawals before 70-1/2. Further, even though IRAs may create a lot of millionaires in the next 35 or 40 years, inflation will still be eroding the buying power of those future dollars.

IRA Planner will help you plot your financial strategy for your retirement years. You tell the TI your age, the size of your annual IRA contribution, and your expectations about future interest and inflation rates, and it tells you what your IRA will be worth at maturity.

Figure 1-3. IRA Planner

```
          I.R.A. PAYOFF THROUGH AGE 70
-------------------------------------------------
                YOUR AGE = 33 YEARS
                IRA SPAN = 37 YEARS

PAYMENTS
                ANNUAL = $ 2000.00
                 TOTAL = $74000.00

IRA PAYOFF
               CURRENT = $1217661.07
              CONSTANT = $ 200226.87
```

For example, suppose you are 33 years old, in a 50 percent tax bracket, and decide to put $2000 into an IRA each year. You expect interest rates to average about 12 percent a year, and you guess that inflation will run about 5 percent per year.

Plugging these figures into the TI, you'll find that your IRA will be worth $1.2 million by the time you're 70 years old. You'll also learn that, in terms of today's dollars, that whopping sum is worth just $200,000, the constant dollar payoff. Nevertheless, your total IRA payments will equal only $74,000, half of which would have gone to the IRS anyway had you not invested in an IRA. In this light, the constant dollar payoff is actually pretty high.

Program 1-3. IRA Planner

```
100 REM I.R.A. PLANNER
130 REM HEADING
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 520
170 REM COMPUTE
180 GOSUB 1220
190 REM DISPLAY RESULTS
200 GOSUB 1340
210 END
220 REM HEADING
230 REM TITLE
240 GOSUB 280
250 REM EXPLANATION
260 GOSUB 360
270 RETURN
280 REM TITLE
290 CALL CLEAR
300 DISPLAY AT(8,8):"Individual"
310 DISPLAY AT(10,9):"Retirement"
320 DISPLAY AT(12,10):"Account"
330 DISPLAY AT(14,11):"Planner"
340 FOR DELAY=1 TO 1000 :: NEXT DELAY
350 RETURN
360 REM EXPLANATION
370 CALL CLEAR
380 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
390 DISPLAY AT(2,1):"VALUE OF YOUR I.R.A. AT"
400 DISPLAY AT(3,1):"AGE 59 AND 70."
410 DISPLAY AT(6,1):"59 IS THE EARLIEST AGE THAT"
420 DISPLAY AT(7,1):"YOU CAN START WITHDRAWING"
430 DISPLAY AT(8,1):"FUNDS WITHOUT PENALTY."
440 DISPLAY AT(11,1):"AND 70 IS THE LATEST AGE"
450 DISPLAY AT(12,1):"THAT YOU CAN DELAY WITH-"
460 DISPLAY AT(13,1):"DRAWING, WITHOUT PENALTY."
470 DISPLAY AT(16,1):"VALUES ARE DISPLAYED IN"
```

```
480 DISPLAY AT(17,1):"CURRENT & CONSTANT DOLLARS."
490 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
500 ACCEPT AT(23,26):Z$
510 RETURN
520 REM ENTER DATA
530 REM SELECT 59 OR 70
540 GOSUB 640
550 REM AGE
560 GOSUB 740
570 REM YEARLY DEPOSIT
580 GOSUB 870
590 REM INTEREST RATE
600 GOSUB 960
610 REM INFLATION RATE
620 GOSUB 1090
630 RETURN
640 REM SELECT 59 OR 79
650 CALL CLEAR
660 DISPLAY AT(1,1):"WOULD YOU LIKE I.R.A."
670 DISPLAY AT(2,1):"COMPUTATIONS MADE THROUGH"
680 DISPLAY AT(5,3):"1. AGE 59, OR"
690 DISPLAY AT(6,3):"2. AGE 70"
700 DISPLAY AT(9,3):"AGE = ?"
710 ACCEPT AT(9,9)BEEP VALIDATE("12",""")SIZE(1):C$
720 IF C$="" THEN 710 ELSE CHOICE=VAL(C$)
730 RETURN
740 REM AGE
750 CALL CLEAR
760 DISPLAY AT(1,1):"PARDON MY ASKING, BUT"
770 DISPLAY AT(2,1):"HOW OLD ARE YOU ?"
780 ACCEPT AT(2,18)BEEP VALIDATE(DIGIT,""):A$
790 IF A$="" THEN 780 ELSE AGE=VAL(A$)
800 IF AGE=0 THEN DISPLAY AT(23,1):"WHO ARE YOU KI
    DDING!" :: GOTO 780
810 IF CHOICE=1 AND AGE>=59 OR(CHOICE=2 AND AGE>=7
    0)THEN GOSUB 830
820 RETURN
830 REM I.R.A. INAPPROPRIATE
840 DISPLAY AT(22,1):"SORRY, BUT AN I.R.A. IS"
850 DISPLAY AT(23,1):"PROBABLY NOT FOR YOU."
860 STOP
870 REM DEPOSIT
880 CALL CLEAR
890 DISPLAY AT(1,1):"HOW MUCH DO YOU WANT TO"
900 DISPLAY AT(2,1):"PUT INTO YOUR I.R.A. EACH"
910 DISPLAY AT(3,1):"YEAR ?"
920 ACCEPT AT(3,8)BEEP VALIDATE(NUMERIC,""):D$
930 IF D$="" THEN 920 ELSE DEPOSIT=VAL(D$)
```

```
940 IF DEPOSIT=0 THEN DISPLAY AT(23,1):"PLEASE DEP
    OSIT SOMETHING !" :: GOTO 920
950 RETURN
960 REM INTEREST RATE
970 CALL CLEAR
980 DISPLAY AT(1,1):"PLEASE ENTER THE INTEREST"
990 DISPLAY AT(2,1):"RATE THAT YOU EXPECT WILL"
1000 DISPLAY AT(3,1):"PREVAIL, ON AVERAGE, OVER"
1010 DISPLAY AT(4,1):"THE LIFE OF YOUR I.R.A."
1020 DISPLAY AT(7,1):"FOR EXAMPLE, ENTER 7 FOR"
1030 DISPLAY AT(8,1):"7%, 10 FOR 10%, & SO ON."
1040 DISPLAY AT(10,1):"ANNUAL RATE = ?"
1050 ACCEPT AT(10,17)BEEP VALIDATE(NUMERIC,""):R$
1060 IF R$="" THEN 1050 ELSE INTRATE=VAL(R$)
1070 IF INTRATE=0 THEN DISPLAY AT(23,1):"THAT'S PR
    ETTY LOW !" :: GOTO 1050
1080 RETURN
1090 REM INFLATION RATE
1100 CALL CLEAR
1110 DISPLAY AT(1,1):"PLEASE ENTER THE INFLATION"
1120 DISPLAY AT(2,1):"RATE THAT YOU EXPECT WILL"
1130 DISPLAY AT(3,1):"PREVAIL, ON AVERAGE, OVER"
1140 DISPLAY AT(4,1):"THE LIFE OF YOUR I.R.A."
1150 DISPLAY AT(7,1):"FOR EXAMPLE, ENTER 5 FOR"
1160 DISPLAY AT(8,1):"5%, 12 FOR 12%, & SO ON."
1170 DISPLAY AT(11,1):"ANNUAL RATE = ?"
1180 ACCEPT AT(11,17)BEEP VALIDATE(NUMERIC,""):F$
1190 IF F$="" THEN 1180 ELSE INFRATE=VAL(F$)
1200 IF INFRATE=0 THEN DISPLAY AT(23,1):"NO SUCH L
    UCK !" :: GOTO 1180
1210 RETURN
1220 REM COMPUTE
1230 IF CHOICE=1 THEN Y=59 ELSE Y=70
1240 REM NUMBER OF YEARS UNTIL MATURITY
1250 N=Y-AGE
1260 REM CURRENT-DOLLAR VALUE
1270 FOR I=1 TO N
1280 VALUE=VALUE+DEPOSIT*(1+INTRATE/100)^(N-I+1)
1290 NEXT I
1300 REM DEFLATE
1310 INDEX=(1+INFRATE/100)^N
1320 KVALUE=VALUE/INDEX
1330 RETURN
1340 REM RESULTS
1350 CALL CLEAR
1360 CALL HCHAR(1,3,61,28)
1370 DISPLAY AT(2,8):"I.R.A.  PAYOFF"
1380 DISPLAY AT(3,8):"THROUGH AGE";Y
1390 CALL HCHAR(4,3,61,28)
```

```
1400 IMAGE = ## YEARS
1410 DISPLAY AT(6,6):"YOUR AGE" :: DISPLAY AT(6,15
     ):USING 1400:AGE
1420 DISPLAY AT(7,3):"I.R.A. SPAN" :: DISPLAY AT(7
     ,15):USING 1400:N
1430 DISPLAY AT(10,6):"PAYMENTS"
1440 IMAGE = $########.##
1450 DISPLAY AT(11,8):"ANNUAL" :: DISPLAY AT(11,15
     ):USING 1440:DEPOSIT
1460 DISPLAY AT(12,9):"TOTAL" :: DISPLAY AT(12,15)
     :USING 1440:N*DEPOSIT
1470 DISPLAY AT(15,1):"I.R.A. PAYOFF"
1480 DISPLAY AT(16,7):"CURRENT" :: DISPLAY AT(16,1
     5):USING 1440:VALUE
1490 DISPLAY AT(17,6):"CONSTANT" :: DISPLAY AT(17,
     15):USING 1440:KVALUE
1500 CALL HCHAR(23,3,61,28)
1510 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1520 ACCEPT AT(24,26):Z$
1530 RETURN
```

Municipal Bond Buyer

If you're like most people, you cringe every April after computing how much of your interest income goes to the IRS. If you're in a 40 percent marginal tax bracket, for example, $400 out of every $1000 of interest that you earn is paid to the Treasury. This hurts. Indeed, making such a payment just once is enough to elicit the plea, "Isn't there a way that I can keep more of my money?"

Fortunately, there is. Municipal bonds offer yields that are not only highly competitive with money-market certificates but free from federal income taxes as well. Further, municipal bonds are usually exempt from state taxes in the state where they were issued. Therefore, for those in a 50 percent tax bracket, a 9 percent return on a municipal bond is equivalent to an 18 percent return on a taxable security. It's no wonder that sales of municipal bonds have been increasing over the past few years.

A municipal bond is a promise to pay issued by a state, city, county, town, housing authority, university, and so on. The organization sells the bond and promises to pay the principal back at a later date, say in 20 or 30 years, with a fixed amount of interest usually paid every six months. The bonds are issued to finance construction of public works projects like

schools, roads, sewers, nursing homes, hospitals, and low-income housing. Why not use current tax revenues to pay for these projects instead? Simply because many of the projects will last a lifetime, and because it's only right that future generations pay their fair share of the cost.

Using this program, your TI calculates key municipal bond values. To illustrate this, suppose that Fall River, Massachusetts, decides to build a sewer. Construction is financed through the sale of Fall River bonds, with each bond carrying a coupon rate of 10 percent and a face value of $1000. The bonds were issued in January 1983 and will mature in November 2004. Interest is paid every six months, beginning in June 1983.

Assume that you purchase ten $1000 slices of the Fall River issue in March 1983, at a price of $103. This price is related to a base price or par value of $100.

After entering the numbers as directed by the program, you will get the display shown in Figure 1-4. It tells you that the total cost of your purchase was $10,466.67. Of that amount, $10,300 was *paid to principal.* That is, 10 bonds x $1000 per bond x 1.03 (price index) = $10,300. An additional $166.67 was paid to interest. But where do these figures come from?

Figure 1-4. Bond Buyer

```
          MUNICIPAL BOND SUMMARY
-------------------------------------------
     COUPON RATE  =   10.00%
      BOND PRICE  =  103.00
   INTEREST DATE  :  12/82
  PURCHASE DATE   :   3/83
  MATURITY DATE   :  11/2004
          LENGTH  :  21 YR 8 MTH

     TOTAL COST
      PRINCIPAL  =  $10300.00
       INTEREST  =  $  166.67
            NET  =  $10466.67

    TOTAL INCOME
      PRINCIPAL  =  $10000.00
       INTEREST  =  $21833.33
            NET  =  $31833.33
```

Note that the interest payment on a bond is *always* fixed. In the Fall River case, you receive $500 in June 1983, $500 in December 1983, $500 in June 1984, $500 in December 1984, and so on. But you purchased your bonds in March 1983. As a result, you owe the seller two months of interest (for January and February). That's 2 x $500/6 (the monthly interest payment), or $166.67.

Press ENTER again, and the TI also tells you that you will earn $21,833.33 in tax-free interest over 21 years and eight months, the life of our bond. The annual current yield on your investment is the coupon rate divided by the price (10/103 = 0.0971), or 9.71 percent. Assuming a 50 percent tax bracket, the equivalent yield on a taxable security is a whopping 19.42 percent per annum. No wonder municipal bonds are becoming so popular.

Program 1-4. Municipal Bond Buyer

```
100 REM MUNICIPAL BOND BUYER
130 REM HEADING
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 290
170 REM COMPUTE
180 GOSUB 1260
190 REM DISPLAY RESULTS
200 GOSUB 1690
210 END
220 REM HEADING
230 CALL CLEAR
240 DISPLAY AT(10,10):"Municipal"
250 DISPLAY AT(12,12):"Bond"
260 DISPLAY AT(14,14):"Buyer"
270 FOR DELAY=1 TO 500 :: NEXT DELAY
280 RETURN
290 REM ENTER DATA
300 REM COUPON RATE
310 GOSUB 410
320 REM PURCHASE PRICE
330 GOSUB 520
340 REM NUMBER OF UNITS PURCHASED
350 GOSUB 680
360 REM DATES
370 GOSUB 770
380 REM TAX BRACKET
390 GOSUB 1140
```

```
400 RETURN
410 REM COUPON RATE
420 CALL CLEAR
430 DISPLAY AT(1,1):"PLEASE ENTER THE COUPON"
440 DISPLAY AT(2,1):"RATE ON YOUR BOND."
450 DISPLAY AT(4,1):"FOR EXAMPLE, ENTER 7 FOR"
460 DISPLAY AT(5,1):"7%, 10 FOR 10%, & SO ON."
470 DISPLAY AT(8,1):"RATE = ?"
480 ACCEPT AT(8,10)BEEP VALIDATE(NUMERIC,""):R$
490 IF R$="" THEN 480 ELSE RATE=VAL(R$)
500 IF RATE<=0 THEN DISPLAY AT(23,1):"THAT'S PRETT
    Y LOW!" :: GOTO 480
510 RETURN
520 REM PRICE
530 CALL CLEAR
540 DISPLAY AT(1,1):"PLEASE ENTER THE PRICE"

550 DISPLAY AT(2,1):"OF YOUR BOND."
560 DISPLAY AT(4,1):"USE $100 AS A BASE FIGURE"
570 DISPLAY AT(5,1):"OR 'PAR' VALUE."
580 DISPLAY AT(7,1):"FOR EXAMPLE, ENTER A NUMBER"
590 DISPLAY AT(8,1):"LIKE $95 ($5.00 DISCOUNT)."
600 DISPLAY AT(10,1):"OR ENTER A NUMBER LIKE"
610 DISPLAY AT(11,1):"$103.75 ($3.75 PREMIUM),"
620 DISPLAY AT(12,1):"AND SO ON."
630 DISPLAY AT(15,1):"PRICE = ?"
640 ACCEPT AT(15,11)BEEP VALIDATE(NUMERIC,""):P$
650 IF P$="" THEN 640 ELSE PRICE=VAL(P$)
660 IF PRICE<=0 OR PRICE>200 THEN DISPLAY AT(23,1)
    :"'PAR' VALUE IS $100.00 !" :: GOTO 640
670 RETURN
680 REM PURCHASE
690 CALL CLEAR
700 DISPLAY AT(1,1):"HOW MANY $1,000 SLICES"
710 DISPLAY AT(2,1):"OF THE BOND DO YOU WANT"
720 DISPLAY AT(3,1):"TO BUY ?"
730 ACCEPT AT(3,10)BEEP VALIDATE(DIGIT,""):Q$
740 IF Q$="" THEN 730 ELSE QTY=VAL(Q$)
750 IF QTY=0 THEN DISPLAY AT(23,1):"PLEASE BUY SOM
    ETHING !" :: GOTO 730
760 RETURN
770 REM DATES
780 REM LAST INTEREST PAYMENT
790 GOSUB 850
800 REM PURCHASE
810 GOSUB 960
820 REM MATURITY
830 GOSUB 1050
840 RETURN
```

```
850 REM LAST INTEREST PAYMENT
860 CALL CLEAR
870 DISPLAY AT(1,1):"PLEASE ENTER THE DATE THAT"
880 DISPLAY AT(2,1):"INTEREST WAS LAST PAID"
890 DISPLAY AT(3,1):"ON YOUR BOND."
900 DISPLAY AT(5,1):"IF INTEREST HAS NOT YET"
910 DISPLAY AT(6,1):"BEEN PAID, DON'T WORRY."
920 DISPLAY AT(8,1):"ENTER THE DATE THAT THE"
930 DISPLAY AT(9,1):"BOND WAS ISSUED INSTEAD."
940 CALL DATE(MT(1),YR(1))
950 RETURN
960 REM PURCHASE DATE
970 CALL CLEAR
980 DISPLAY AT(2,1):"PLEASE ENTER THE DATE THAT"
990 DISPLAY AT(3,1):"YOU PURCHASED THE BOND."
1000 CALL DATE(MT(2),YR(2))
1010 REM MONTHS BETWEEN DATES
1020 N1=(YR(2)-YR(1))*12+MT(2)-MT(1)
1030 IF N1<0 THEN DISPLAY AT(23,1):"PLEASE ENTER A
      LATER DATE" :: DISPLAY AT(13,11):"" :: GOTO
      1000
1040 RETURN
1050 REM MATURITY
1060 CALL CLEAR
1070 DISPLAY AT(2,1):"PLEASE ENTER THE DATE"
1080 DISPLAY AT(3,1):"THAT YOUR BOND MATURES."
1090 CALL DATE(MT(3),YR(3))
1100 REM MONTHS BETWEEN DATES
1110 N2=(YR(3)-YR(2))*12+MT(3)-MT(2)
1120 IF N2<0 THEN DISPLAY AT(23,1):"PLEASE ENTER A
      LATER DATE" :: DISPLAY AT(13,11):"" :: GOTO
      1090
1130 RETURN
1140 REM TAX BRACKET
1150 CALL CLEAR
1160 DISPLAY AT(1,1):"PLEASE ENTER YOUR MARGINAL"
1170 DISPLAY AT(2,1):"TAX BRACKET."
1180 DISPLAY AT(5,1):"FOR EXAMPLE, ENTER 33 IF"
1190 DISPLAY AT(6,1):"YOU'RE IN THE 33% BRACKET."
1200 DISPLAY AT(9,1):"AND ENTER 50 IF YOU'RE IN"
1210 DISPLAY AT(10,1):"THE 50% BRACKET, & SO ON."
1220 DISPLAY AT(15,1):"BRACKET = ?"
1230 ACCEPT AT(15,13)BEEP VALIDATE(DIGIT,"")SIZE(2
     ):B$
1240 IF B$="" THEN 1230 ELSE BK=VAL(B$)
1250 RETURN
1260 REM COMPUTE
1270 REM TOTAL COST
1280 GOSUB 1360
```

```
1290 REM TIME TO MATURITY
1300 GOSUB 1470
1310 REM TOTAL PAYMENT
1320 GOSUB 1510
1330 REM YIELDS
1340 GOSUB 1560
1350 RETURN
1360 REM COST
1370 RATE=RATE/100
1380 REM MONTHLY INTEREST PAYMENT
1390 MPAY=QTY*1000*RATE/12
1400 REM INTEREST COST
1410 INTCOST=MPAY*N1
1420 REM PRINCIPAL COST
1430 PRNCOST=QTY*PRICE*10
1440 REM NET
1450 NETCOST=INTCOST+PRNCOST
1460 RETURN
1470 REM TIME TO MATURITY
1480 YEARS=INT(N2/12)
1490 MTHS=N2-YEARS*12
1500 RETURN
1510 REM TOTAL PAYMENTS
1520 PRNPAY=QTY*1000
1530 INTPAY=PRNPAY*RATE*(N1+N2)/12
1540 NETPAY=PRNPAY+INTPAY
1550 RETURN
1560 REM YIELDS
1570 REM TAX-FREE
1580 GOSUB 1620
1590 REM TAXABLE
1600 GOSUB 1660
1610 RETURN
1620 REM TAX-FREE
1630 REM CURRENT
1640 CURRYD=RATE*100/PRICE
1650 RETURN
1660 REM TAXABLE YIELDS
1670 TCURRYD=CURRYD/(1-BK/100)
1680 RETURN
1690 REM DISPLAY RESULTS
1700 REM SUMMARY
1710 GOSUB 1750
1720 REM YIELDS
1730 GOSUB 2040
1740 RETURN
1750 REM SUMMARY
1760 CALL CLEAR
```

```
177Ø CALL HCHAR(1,3,61,28)
178Ø DISPLAY AT(2,4):"MUNICIPAL BOND SUMMARY"
179Ø CALL HCHAR(3,3,61,28)
18ØØ IMAGE = ####.## %
181Ø IMAGE = $###.##
182Ø DISPLAY AT(5,1):"Coupon Rate" :: DISPLAY AT(5
     ,15):USING 18ØØ:RATE*1ØØ
183Ø DISPLAY AT(6,1):"Bond Price" :: DISPLAY AT(6,
     15):USING 181Ø:PRICE
184Ø IMAGE : ##/####
185Ø DISPLAY AT(8,1):"Interest Date" :: DISPLAY AT
     (8,15):USING 184Ø:MT(1),YR(1)
186Ø DISPLAY AT(9,1):"Purchase Date" :: DISPLAY AT
     (9,15):USING 184Ø:MT(2),YR(2)
187Ø DISPLAY AT(1Ø,1):"Maturity Date" :: DISPLAY A
     T(1Ø,15):USING 184Ø:MT(3),YR(3)
188Ø DISPLAY AT(11,8):"Length"
189Ø IMAGE : ##yrs ##mths
19ØØ DISPLAY AT(11,15):USING 189Ø:YEARS,MTHS
191Ø DISPLAY AT(13,1):"Total Cost"
192Ø IMAGE = $########.##
193Ø DISPLAY AT(14,2):"Principal" :: DISPLAY AT(14
     ,15):USING 192Ø:PRNCOST
194Ø DISPLAY AT(15,2):"Interest" :: DISPLAY AT(15,
     15):USING 192Ø:INTCOST
195Ø DISPLAY AT(16,2):"Net" :: DISPLAY AT(16,15):U
     SING 192Ø:NETCOST
196Ø DISPLAY AT(18,1):"Total Income"
197Ø DISPLAY AT(19,2):"Principal" :: DISPLAY AT(19
     ,15):USING 192Ø:PRNPAY
198Ø DISPLAY AT(2Ø,2):"Interest" :: DISPLAY AT(2Ø,
     15):USING 192Ø:INTPAY
199Ø DISPLAY AT(21,2):"Net" :: DISPLAY AT(21,15):U
     SING 192Ø:NETPAY
2ØØØ CALL HCHAR(23,3,61,28)
2Ø1Ø DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
2Ø2Ø ACCEPT AT(24,26):Z$
2Ø3Ø RETURN
2Ø4Ø REM YIELDS
2Ø5Ø CALL CLEAR
2Ø6Ø CALL HCHAR(1,3,61,28)
2Ø7Ø DISPLAY AT(2,5):"MUNICIPAL BOND YIELDS"
2Ø8Ø CALL HCHAR(3,3,61,28)
2Ø9Ø DISPLAY AT(5,1):"TAX-FREE"
21ØØ IMAGE = ###.## %
2111 DISPLAY AT(6,2):"Current" :: DISPLAY AT(6,16)
     :USING 21ØØ:CURRYD*1ØØ
2120 DISPLAY AT(1Ø,1):"TAX-EQUIVALENT"
```

```
2130 DISPLAY AT(11,2):"Current" :: DISPLAY AT(11,1
     6):USING 2100:TCURRYD*100
2140 CALL HCHAR(23,3,61,28)
2150 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
2160 ACCEPT AT(24,26):Z$
2170 RETURN
2180 SUB DATE(M,Y)
2190 DISPLAY AT(12,4):"YEAR ="
2200 ACCEPT AT(12,11)BEEP VALIDATE(DIGIT,"")SIZE(4
     ):Y$
2210 IF Y$="" THEN 2200 ELSE Y=VAL(Y$)
2220 IF LEN(Y$)<4 THEN DISPLAY AT(23,1):"PLEASE EN
     TER A 4-DIGIT YEAR" :: GOTO 2200
2230 DISPLAY AT(23,1):""
2240 DISPLAY AT(13,3):"MONTH ="
2250 ACCEPT AT(13,11)BEEP VALIDATE(DIGIT,"")SIZE(2
     ):M$
2260 IF M$="" THEN 2250 ELSE M=VAL(M$)
2270 S$="BAD NUMBER"
2280 FOR J=1 TO 12
2290 IF M=J THEN S$="GOOD NUMBER"
2300 NEXT J
2310 IF S$="BAD NUMBER" THEN DISPLAY AT(23,1):"PLE
     ASE ENTER 1 TO 12" :: GOTO 2250
2320 SUBEND
```

Loan Payment

After many months of soul-searching, you finally succumb to your long-cherished dream—owning that high-priced, envy-inducing paragon of conspicuous consumption, a new French sports car.

"Price is no object," you tell the dealer. And then....

"You bought *what?*"

"Just a little bitty sports car...."

"But how much did you *pay* for it?"

How much *did* you pay for it? You can use this TI computer program to find out. Assume that you borrowed $12,000 to pay for the car, and that you'll be paying back your loan over the next five-and-a-half years. Following the prompts, enter $12,000 for the amount of the loan, five years and six months for the length of the loan, and 12 percent for the interest rate on the loan. The display then tells you that your monthly payment is $249.25 and that total payments equal $16,450.25. Of this amount, $12,000 is paid to principal, while $4,450.25 goes to interest.

Figure 1-5. Loan Payment

SUMMARY OF THE LOAN

LOAN:		
AMOUNT	=	$12000.00
INT. RATE	=	12.00%
LENGTH	=	5 YR 6 MTH
PAYMENTS:		
MONTHLY	=	$ 249.25
TOTAL	=	$16450.25
PRINCIPAL	=	$12000.00
INTEREST	=	$ 4450.25

The monthly payment on a loan is calculated using the following formula:

$$\text{Payment} = L*R*(1+R)^N/[(1+R)^N - 1]$$

where L = amount of the loan
R = *monthly* interest rate in decimal form. For example, 12 percent per year becomes 0.12/12 = 0.01
N = number of months over the life the loan

Program 1-5. Loan Payment

```
100 REM LOAN PAYMENTS
130 REM ENTER DATA
140 GOSUB 200
150 REM COMPUTE
160 GOSUB 670
170 REM DISPLAY RESULTS
180 GOSUB 820
190 END
200 REM ENTER DATA
210 CALL CLEAR
220 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES"
230 DISPLAY AT(2,1):"LOAN PAYMENTS."
240 REM ENTER AMOUNT OF LOAN
250 GOSUB 310
260 REM ENTER LENGTH OF LOAN
270 GOSUB 390
280 REM ENTER INTEREST RATE
290 GOSUB 560
```

```
300 RETURN
310 REM AMOUNT OF LOAN
320 DISPLAY AT(5,1):"HOW MUCH MONEY WOULD YOU"
330 DISPLAY AT(6,1):"LIKE TO BORROW ?"
340 ACCEPT AT(6,17)BEEP VALIDATE(NUMERIC,""):L$
350 IF L$="" THEN 340 ELSE LOAN=VAL(L$)
360 IF LOAN<=0 THEN DISPLAY AT(23,1):"PLEASE BORRO
    W SOMETHING !" :: GOTO 340
370 IF LOAN>9999999 THEN DISPLAY AT(23,1):"PLEASE
    SCALE DOWN FIGURE" :: GOTO 340
380 RETURN
390 REM LENGTH OF LOAN
400 CALL CLEAR
410 DISPLAY AT(1,1):"PLEASE ENTER THE LENGTH OF"
420 DISPLAY AT(2,1):"YOUR LOAN IN YEARS AND"
430 DISPLAY AT(3,1):"MONTHS."
440 DISPLAY AT(6,2):"YEARS = ?"
450 ACCEPT AT(6,12)BEEP VALIDATE(DIGIT,""):Y$
460 IF Y$="" THEN 450 ELSE Y=VAL(Y$)
470 DISPLAY AT(8,1):"MONTHS = ?"
480 ACCEPT AT(8,12)BEEP VALIDATE(DIGIT,"")SIZE(2):
    M$
490 IF M$="" THEN 480 ELSE M=VAL(M$)
500 S$="TRY AGAIN"
510 FOR J=0 TO 12
520 IF J=M THEN S$="OKAY"
530 NEXT J
540 IF S$="TRY AGAIN" THEN DISPLAY AT(23,1):"PLEAS
    E ENTER 1 TO 12" :: GOTO 480
550 RETURN
560 REM INTEREST RATE
570 CALL CLEAR
580 DISPLAY AT(1,1):"PLEASE ENTER THE INTEREST"
590 DISPLAY AT(2,1):"RATE ON YOUR LOAN."
600 DISPLAY AT(4,1):"FOR EXAMPLE, ENTER 8 FOR"
610 DISPLAY AT(5,1):"8%, 11 FOR 11%, AND SO ON."
620 DISPLAY AT(9,1):"INTEREST RATE = ?"
630 ACCEPT AT(9,19)BEEP VALIDATE(NUMERIC,""):R$
640 IF R$="" THEN 630 ELSE RATE=VAL(R$)
650 IF RATE<=0 THEN DISPLAY AT(23,1):"THERE'S NO S
    UCH THING AS A" :: DISPLAY AT(24,1):"FREE LUNCH
    !" :: GOTO 630
660 RETURN
670 REM COMPUTE
680 REM TOTAL NUMBER OF MONTHS
690 N=Y*12+M
700 REM INTEREST RATE PER PERIOD
710 R=RATE/100/12
720 REM PAYMENT PER PERIOD
```

```
730 REM NUMERATOR
740 NU=LOAN*R*(1+R)^N
750 REM DENOMINATOR
760 DE=(1+R)^N-1
770 REM QUOTIENT
780 PPP=NU/DE
790 REM TOTAL PAYMENT
800 TPAYMENT=N*PPP
810 RETURN
820 REM DISPLAY
830 CALL CLEAR
840 IMAGE = $#########.##
850 IMAGE =  #########.##
860 IMAGE =  #########.##%
870 IMAGE =### Yrs ## Mths
880 CALL HCHAR(1,3,61,28)
890 DISPLAY AT(2,5):"SUMMARY OF THE LOAN"
900 CALL HCHAR(3,3,61,28)
910 DISPLAY AT(5,1):"LOAN:"
920 DISPLAY AT(7,2):"Amount" :: DISPLAY AT(7,13):U
    SING 840:LOAN
930 DISPLAY AT(8,2):"Int. Rate" :: DISPLAY AT(8,13
    ):USING 860:RATE
940 DISPLAY AT(10,2):"Length" :: DISPLAY AT(10,13)
    :USING 870:Y,M
950 DISPLAY AT(14,1):"PAYMENTS:"
960 DISPLAY AT(16,2):"Monthly" :: DISPLAY AT(16,13
    ):USING 840:PPP
970 DISPLAY AT(18,2):"Total" :: DISPLAY AT(18,13):
    USING 840:TPAYMENT
980 DISPLAY AT(20,3):"Principal" :: DISPLAY AT(20,
    13):USING 840:LOAN
990 DISPLAY AT(21,3):"Interest" :: DISPLAY AT(21,1
    3):USING 840:TPAYMENT-LOAN
1000 CALL HCHAR(23,3,61,28)
1010 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1020 ACCEPT AT(24,26):Z$
1030 RETURN
```

Mortgage Payment

Mortgage Payment computes annual, quarterly, or monthly payments on a mortgage. The routine works much like the previous one, but it features some refinements to deal specifically with mortgage-related questions.

To use the program, enter the amount of money to borrow, the length of the loan, and the yearly interest rate. The

TI then outputs your annual, quarterly, or monthly payment, whichever you choose, as well as the total payment over the life of the loan. It also tells you how much of each payment goes toward the principal and how much is interest. This is important because mortgage interest payments can be deducted from gross income, dollar for dollar, in computing income taxes.

To see how the program works, suppose you borrow $50,000, at 12.5 percent interest, to buy a new house. You plan to make monthly payments over a 30-year period. Enter this data into your TI. The computer then tells you that your monthly payment will be $533.63. Total payments over the life of the loan are $192,106.40, with $50,000 paid to principal and $142,106.40 paid to interest.

Figure 1-6. Mortgage Payments

SUMMARY OF THE LOAN

LOAN PARAMETERS:

AMOUNT	=	$ 50000.00
NO. OF YEARS	=	$ 30.00
PAYMENTS/YR.	=	$ 12.00
INT. RATE	=	12.50%

LOAN PAYMENTS:

TOTAL	=	$192106.40
PRINCIPAL	=	$ 50000.00
INTEREST	=	$142106.40

The TI also displays how much of *each* monthly payment goes to principal and interest, as Figure 1-7 shows.

Figure 1-7. Division of Monthly Payments

DIVISION OF MONTHLY PAYMENTS

YEAR:MONTH	PAID TO PRINCIPAL	PAID TO INTEREST
1:1	$ 12.80	$520.83
1:2	12.93	520.70
1:3	13.06	520.57
.	.	.
.	.	.
.	.	.
30:12	528.13	5.50

Program 1-6. Mortgage Payments

```
100 REM MORTGAGE PAYMENTS
130 REM ENTER DATA
140 GOSUB 200
150 REM COMPUTE
160 GOSUB 710
170 REM DISPLAY RESULTS
180 GOSUB 900
190 END
200 REM ENTER DATA
210 CALL CLEAR
220 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES"
230 DISPLAY AT(2,1):"MORTGAGE PAYMENTS."
240 REM ENTER AMOUNT OF LOAN
250 GOSUB 330
260 REM ENTER NUMBER OF YEARS
270 GOSUB 410
280 REM ENTER NUMBER OF PAYMENTS PER YEAR
290 GOSUB 500
300 REM ENTER INTEREST RATE
310 GOSUB 600
320 RETURN
330 REM AMOUNT OF LOAN
340 DISPLAY AT(5,1):"HOW MUCH MONEY WOULD YOU"
350 DISPLAY AT(6,1):"LIKE TO BORROW ?"
360 ACCEPT AT(6,17)BEEP VALIDATE(NUMERIC,""):L$
370 IF L$="" THEN 360 ELSE LOAN=VAL(L$)
380 IF LOAN<=0 THEN DISPLAY AT(23,1):"PLEASE BORRO
    W SOMETHING !" :: GOTO 360
390 IF LOAN>999999 THEN DISPLAY AT(23,1):"PLEASE S
    CALE DOWN FIGURE" :: GOTO 360
400 RETURN
410 REM NUMBER OF YEARS
420 CALL CLEAR
430 DISPLAY AT(1,1):"HOW MANY YEARS IS"
440 DISPLAY AT(2,1):"YOUR LOAN FOR ?"
450 ACCEPT AT(2,17)BEEP VALIDATE(NUMERIC,""):Y$
460 IF Y$="" THEN 450 ELSE Y=VAL(Y$)
470 IF Y<=0 THEN DISPLAY AT(23,1):"THAT'S A SHORT
    LOAN!" :: GOTO 450
480 IF Y>150 THEN DISPLAY AT(23,1):"YOU WON'T LIVE
     THAT LONG!" :: GOTO 450
490 RETURN
500 REM PAYMENTS PER YEAR
510 CALL CLEAR
520 DISPLAY AT(1,1):"PAYMENTS CAN BE MADE EVERY:"
530 DISPLAY AT(4,4):"1. YEAR"
540 DISPLAY AT(5,4):"2. QUARTER, OR"
```

```
550 DISPLAY AT(6,4):"3. MONTH."
560 DISPLAY AT(9,1):"WHICH WOULD YOU LIKE ?"
570 ACCEPT AT(9,24)BEEP VALIDATE("123",""):C$
580 IF C$="" THEN 570 ELSE CHOICE=VAL(C$)
590 RETURN
600 REM INTEREST RATE
610 CALL CLEAR
620 DISPLAY AT(1,1):"PLEASE ENTER THE INTEREST"
630 DISPLAY AT(2,1):"RATE ON YOUR LOAN."
640 DISPLAY AT(4,1):"FOR EXAMPLE, ENTER 8 FOR"
650 DISPLAY AT(5,1):"8%, 11 FOR 11%, AND SO ON."
660 DISPLAY AT(9,1):"INTEREST RATE = ?"
670 ACCEPT AT(9,19)BEEP VALIDATE(NUMERIC,""):R$
680 IF R$="" THEN 670 ELSE RATE=VAL(R$)
690 IF RATE<=0 THEN DISPLAY AT(23,1):"THERE'S NO S
    UCH THING AS A" :: DISPLAY AT(24,1):"FREE LUNC
    H!" :: GOTO 670
700 RETURN
710 REM COMPUTE
720 REM NUMBER OF PAYMENTS PER YEAR
730 IF CHOICE=1 THEN FREQ=1
740 IF CHOICE=2 THEN FREQ=4
750 IF CHOICE=3 THEN FREQ=12
760 REM TOTAL NUMBER OF PAYMENTS
770 N=Y*FREQ
780 REM INTEREST RATE PER PERIOD
790 II=(RATE/100)/FREQ
800 REM PAYMENT PER PERIOD
810 REM NUMERATOR
820 NU=LOAN*II*(1+II)^N
830 REM DENOMINATOR
840 DE=(1+II)^N-1
850 REM QUOTIENT
860 PPP=NU/DE
870 REM TOTAL PAYMENT
880 TPAYMENT=N*PPP
890 RETURN
900 REM DISPLAY RESULTS
910 REM SUMMARY
920 GOSUB 960
930 REM PAYMENTS PER PERIOD
940 GOSUB 1170
950 RETURN
960 REM SUMMARY
970 CALL CLEAR
980 IMAGE = $########.##
990 IMAGE =  ########.##
1000 IMAGE =  ########.##%
1010 CALL HCHAR(1,3,61,28)
```

```
1020 DISPLAY AT(2,5):"SUMMARY OF THE LOAN"
1030 CALL HCHAR(3,3,61,28)
1040 DISPLAY AT(5,1):"LOAN" :: DISPLAY AT(6,1):"PA
     RAMETERS:"
1050 DISPLAY AT(8,7):"Amount" :: DISPLAY AT(8,14):
     USING 980:LOAN
1060 DISPLAY AT(9,1):"No. of Years" :: DISPLAY AT(
     9,14):USING 990:Y
1070 DISPLAY AT(10,1):"Payments/Yr." :: DISPLAY AT
     (10,14):USING 990:FREQ
1080 DISPLAY AT(11,4):"Int. Rate" :: DISPLAY AT(11
     ,14):USING 1000:RATE
1090 DISPLAY AT(15,1):"LOAN PAYMENTS:"
1100 DISPLAY AT(17,8):"Total" :: DISPLAY AT(17,14)
     :USING 980:TPAYMENT
1110 DISPLAY AT(18,4):"Principal" :: DISPLAY AT(18
     ,14):USING 980:LOAN
1120 DISPLAY AT(19,5):"Interest" :: DISPLAY AT(19,
     14):USING 980:TPAYMENT-LOAN
1130 CALL HCHAR(23,3,61,28)
1140 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1150 ACCEPT AT(24,26):Z$
1160 RETURN
1170 REM PAYMENTS PER PERIOD
1180 REM IN BLOCKS OF 12
1190 IF CHOICE=1 THEN P$="YR"
1200 IF CHOICE=2 THEN P$="YR:QT"
1210 IF CHOICE=3 THEN P$="YR:MT"
1220 YR=1 :: QT=1 :: MT=1
1230 FOR I=1 TO N STEP 12
1240 REM HEADING
1250 GOSUB 1300
1260 REM BODY
1270 GOSUB 1420
1280 NEXT I
1290 RETURN
1300 REM HEADING
1310 CALL CLEAR
1320 CALL HCHAR(1,3,61,28)
1330 IF CHOICE=1 THEN DISPLAY AT(2,5):"YEARLY PAYM
     ENT EQUALS"
1340 IF CHOICE=2 THEN DISPLAY AT(2,3):"QUARTERLY P
     AYMENT EQUALS"
1350 IF CHOICE=3 THEN DISPLAY AT(2,4):"MONTHLY PAY
     MENT EQUALS"
1360 IMAGE $######.##
1370 DISPLAY AT(3,10):USING 1360:PPP
1380 CALL HCHAR(4,3,61,28)
1390 DISPLAY AT(6,9):"PAID TO";TAB(22);"PAID TO"
```

```
1400 DISPLAY AT(7,1):P$;TAB(8);"PRINCIPAL";TAB(21)
     ;"INTEREST"
1410 RETURN
1420 REM BODY
1430 IMAGE ##:## $######.##  $######.##
1440 IMAGE ##:#  $######.##  $######.##
1450 IMAGE ##{4 SPACES}$######.##  $######.##
1460 ROW=9
1470 FOR J=I TO I+11
1480 IF J>N THEN 1550
1490 INTR=II*LOAN
1500 PRN=PPP-INTR
1510 REM FREQUENCY
1520 ON CHOICE GOSUB 1600,1640,1680
1530 ROW=ROW+1
1540 LOAN=LOAN-PRN
1550 NEXT J
1560 CALL HCHAR(23,3,61,28)
1570 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1580 ACCEPT AT(24,26):Z$
1590 RETURN
1600 REM YEAR
1610 DISPLAY AT(ROW,1):USING 1450:YR,PRN,INTR
1620 YR=YR+1
1630 RETURN
1640 REM QUARTER
1650 DISPLAY AT(ROW,1):USING 1440:YR,QT,PRN,INTR
1660 IF QT=4 THEN ROW=ROW+1 :: YR=YR+1 :: QT=1 ELS
     E QT=QT+1
1670 RETURN
1680 REM MONTH
1690 DISPLAY AT(ROW,1):USING 1430:YR,MT,PRN,INTR
1700 IF MT=12 THEN YR=YR+1 :: MT=1 ELSE MT=MT+1
1710 RETURN
```

Chapter 2

Basics for Business

Basics for Business

Today's business world is far more complex than the business world of two or three decades ago, but your TI computer can help you comprehend it. This chapter presents a series of programs designed to help corporate executives, small-firm planners, and self-employed entrepreneurs manage their financial resources. You may also want to use some of these routines to help manage your personal finances.

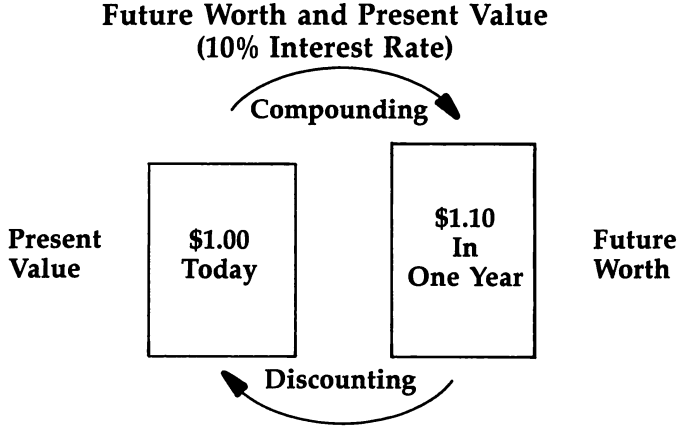
The programs cover a variety of subjects:

- "Net Present Value of a Cash Flow" determines how much a future stream of dollars is worth today.
- "Internal Rate of Return" evaluates investment proposals.
- "Least-Squares Forecasting" uses simple linear-regression analysis to predict the future value of a variable.
- "Time-Series Forecasting" predicts the future value of a variable using extrapolation techniques.
- "Computer Cash Register" turns your TI into a money-counting machine.

Net Present Value of a Cash Flow

Money can have a changing value over time. Given a 10 percent interest rate and annual compounding, $1.00 today will be worth $1.10 in a year. Conversely, next year's $1.10 is worth only $1.00 today. Future worth is computed in the first case, and present value is computed in the second. The two concepts, as Figure 2-1 shows, are exact inverses of each other.

Figure 2-1. Net Present Value



This program computes the net present value of a cash flow for any interest rate and for several frequencies of interest compounding. Net present value tells you how much a future stream of dollars is worth today. This information is valuable if you are forced to choose between alternative investments that pay off over different periods.

To illustrate the calculation, suppose a firm buys a new machine for $1500. The machine is expected to save the firm $3000 a year for two years, the life of the machine.

Initial	Net Revenue	
Cost	Year 1	Year 2
$1500	$3000	$3000

But also suppose that, instead of buying the machine, the firm has the option of investing in a money market certificate and earning a 10 percent return. The discount rate in this case, or the interest rate used in computing net present value, is 10 percent per annum. This percentage is the next-best rate of return on the firm's investment.

The net present value of the firm's cash flow is:

$$NPV = -C + R_1/(1 + r) + R_2(1 + r)^2$$

where C is initial cost, R_1 and R_2 are net revenues in years 1 and 2, and r is the discount rate in decimal form. Hence, NPV = −$1500 + $3000/(1.10) + $3000(1.21), or roughly $3707. The negative sign is attached to initial cost because this figure is actually a negative revenue. Further, net revenue in the first year (R_1) is discounted by 1.10 since X dollars today will be worth X*1.10 dollars in a year. Similarly, net revenue in the second year (R_2) is discounted by 1.21, since X dollars today will be worth X*1.21 dollars in two years.

Figure 2-2. Net Present Value of a Cash Flow

```
INVESTMENT RESULTS

INITIAL COST = $1500.00
INTEREST RATE = 10.00%
NET PRESENT VALUE = $3706.61
```

Enter the data, run the program, and you'll get the output shown in Figure 2-2. In this case, the net present value is $3706.61. A net present value greater than zero means that a proposed investment is economically worthwhile. One less than zero, on the other hand, implies an investment that is

sheer folly. In this situation, the alternative investment (whose yield is represented by the discount rate) will always pay more.

Program 2-1. Net Present Value of a Cash Flow

```
100 REM NET PRESENT VALUE
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 670
170 REM COMPUTE
180 GOSUB 970
190 REM DISPLAY RESULTS
200 GOSUB 1090
210 END
220 REM INITIALIZE
230 REM HEADING
240 GOSUB 300
250 REM INTEREST RATE
260 GOSUB 420
270 REM ENTER FREQUENCY OF INTEREST COMPOUNDING
280 GOSUB 550
290 RETURN
300 REM HEADING
310 DIM R(200)
320 CALL CLEAR
330 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
340 DISPLAY AT(2,1):"NET PRESENT VALUE OF A"
350 DISPLAY AT(3,1):"PROPOSED INVESTMENT."
360 DISPLAY AT(6,1):"HOW MANY PERIODS ARE"
370 DISPLAY AT(7,1):"IN YOUR CASH FLOW ?"
380 ACCEPT AT(7,20)BEEP VALIDATE(DIGIT,""):N$
390 IF N$="" THEN 380 ELSE N=VAL(N$)
400 IF N=0 THEN DISPLAY AT(23,1):"YOU NEED AT LEAS
    T 1 PERIOD" :: GOTO 380
410 RETURN
420 REM INTEREST RATE
430 CALL CLEAR
440 DISPLAY AT(1,1):"PLEASE ENTER THE NOMINAL"
450 DISPLAY AT(2,1):"INTEREST RATE THAT YOU"
460 DISPLAY AT(3,1):"WOULD LIKE TO USE IN"
470 DISPLAY AT(4,1):"COMPUTING NET PRESENT VALUE."
480 DISPLAY AT(6,1):"FOR EXAMPLE, IF THE INTEREST"
490 DISPLAY AT(7,1):"RATE IS 7% A YEAR, ENTER 7."
500 DISPLAY AT(10,1):"RATE = ?"
510 ACCEPT AT(10,10)BEEP VALIDATE(NUMERIC,""):S$
520 IF S$="" THEN 510 ELSE RATE=VAL(S$)
```

```
530 IF RATE<=Ø THEN DISPLAY AT(23,1):"NO FREE LUNC
    H HERE !" :: GOTO 51Ø
540 RETURN
550 REM FREQUENCY OF COMPOUNDING
560 CALL CLEAR
570 DISPLAY AT(1,1):"THE FOLLOWING FREQUENCIES"
580 DISPLAY AT(2,1):"OF INTEREST COMPOUNDING"
590 DISPLAY AT(3,1):"ARE ALLOWED:"
600 DISPLAY AT(6,3):"1. ANNUAL"
610 DISPLAY AT(7,3):"2. QUARTERLY, OR"
620 DISPLAY AT(8,3):"3. CONTINUOUSLY."
630 DISPLAY AT(12,1):"WHICH WOULD YOU LIKE ?"
640 ACCEPT AT(12,24)BEEP SIZE(1)VALIDATE("123","")
    :F$
650 IF F$="" THEN 640 ELSE F=VAL(F$)
660 RETURN
670 REM DATA
680 REM INITIAL COST
690 GOSUB 730
700 REM OTHER PERIODS
710 GOSUB 840
720 RETURN
730 REM INITIAL COST
740 CALL CLEAR
750 DISPLAY AT(1,1):"PLEASE ENTER THE INITIAL"
760 DISPLAY AT(2,1):"COST OF YOUR INVESTMENT."
770 DISPLAY AT(4,1):"FOR EXAMPLE, IF A NEW"
780 DISPLAY AT(5,1):"MACHINE COSTS $10,000 ,"
790 DISPLAY AT(6,1):"ENTER 10000."
800 DISPLAY AT(8,1):"COST = ?"
810 ACCEPT AT(8,10)BEEP VALIDATE(NUMERIC,""):V$
820 IF V$="" THEN 810 ELSE R(Ø)=VAL(V$)
830 RETURN
840 REM DATA
850 CALL CLEAR
860 DISPLAY AT(1,1):"PLEASE ENTER THE NET REVENUE"
870 DISPLAY AT(2,1):"EXPECTED IN EACH PERIOD OF"
880 DISPLAY AT(3,1):"YOUR CASH FLOW."
890 DISPLAY AT(6,1):"'NET' REVENUE IS TOTAL"
900 DISPLAY AT(7,1):"REVENUE MINUS TOTAL COST."
910 FOR I=1 TO N
920 DISPLAY AT(11,1):"PERIOD #";I;TAB(13);"= ?"
930 ACCEPT AT(11,17)VALIDATE(NUMERIC,""):V$
940 IF V$="" THEN 930 ELSE R(I)=VAL(V$)
950 NEXT I
960 RETURN
970 REM COMPUTE
980 NPV=-R(Ø)
990 RATE=RATE/100
```

```
1000 REM DISCOUNT FACTOR
1010 IF F=1 THEN DF=(1+RATE)
1020 IF F=2 THEN DF=(1+RATE/4)^4
1030 IF F=3 THEN DF=EXP(RATE)
1040 REM COMPUTATION
1050 FOR I=1 TO N
1060 NPV=NPV+R(I)/DF^I
1070 NEXT I
1080 RETURN
1090 REM DISPLAY
1100 CALL CLEAR
1110 CALL HCHAR(1,3,61,28)
1120 DISPLAY AT(2,6):"INVESTMENT RESULTS"
1130 CALL HCHAR(3,3,61,28)
1140 IMAGE = $########.##
1150 IMAGE ={7 SPACES}###.##%
1160 DISPLAY AT(5,1):"INITIAL COST"
1170 DISPLAY AT(5,14):USING 1140:R(0)
1180 DISPLAY AT(6,1):"INTEREST RATE"
1190 DISPLAY AT(6,14):USING 1150:RATE*100
1200 DISPLAY AT(9,1):"NET PRESENT"
1210 DISPLAY AT(10,1):"VALUE"
1220 DISPLAY AT(10,14):USING 1140:NPV
1230 CALL HCHAR(23,3,61,28)
1240 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1250 ACCEPT AT(24,26):Z$
1260 RETURN
```

Internal Rate of Return

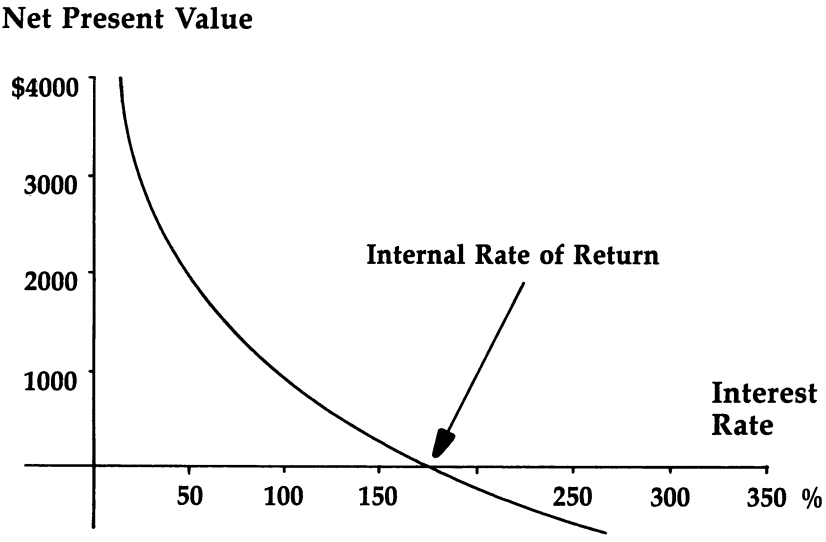
The rational business executive will invest in a proposed project only if it is expected to generate more revenue than expense. In applying this profit-maximizing rule, managers are faced with the problem of linking dollars today and tomorrow. This is because investment projects often generate income for a long period, say for 10 or 20 years, while the bulk of the cost is usually incurred at start-up time.

As noted in the preceding section, net present value (NPV) indicates how much a future stream of dollars is worth today. NPV is tallied by discounting future net revenues, or revenues minus costs, using an interest rate that represents the company's cost of borrowing money.

In the previous example, initial cost was $1500 while net revenue in each of the first and second years was $3000. Using an interest rate of 10 percent, NPV was roughly $3707.

But what if the interest rate changes? For a 20 percent discount rate, NPV would be only $3083. In fact, NPV always falls as the discount rate rises, as Figure 2-3 shows.

Figure 2-3. Net Present Value



In this case, NPV will be zero when the discount rate is 173.21 percent. This percentage is called the Internal Rate of Return, or IRR. It is that interest rate which makes the net present value of a cash flow equal to zero.

Using this program, your TI can calculate IRR values for you. If you use the data from this example, you'll get the output shown in Figure 2-4. An IRR greater than the market rate of interest, a common proxy for the company's cost of borrowing money, means that the investment is profitable. However, an IRR less than the market rate of interest means that the proposed investment should not be considered.

Figure 2-4. Internal Rate of Return

```
INVESTMENT RESULTS

THE INTERNAL RATE OF RETURN
ON YOUR INVESTMENT
IS 173.21%
```

When using this program, note that Newton's method of finding the root of a polynomial is used in tallying the IRR. This requires a guess of the true root, or IRR, which the computer makes for you. The more accurate the initial guess, the quicker the algorithm will find the IRR. In some cases, however, the computer's initial guesses (ten in all) will fail to discover the IRR. When this happens, the TI will instruct you to double the value of the initial guess (line 840). In addition, you might want to try doubling the value of the increment for successive guesses (line 860).

Program 2-2. Internal Rate of Return

```
100 REM IRR
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 370
170 REM COMPUTE
180 GOSUB 780
190 REM DISPLAY RESULTS
200 IF F$="GO!" THEN GOSUB 1420 ELSE GOSUB 1530
210 END
220 REM INITIALIZE
230 DIM R(150),DERV(150),POLY(150)
240 CALL CLEAR
250 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
260 DISPLAY AT(2,1):"INTERNAL RATE OF RETURN"
270 DISPLAY AT(3,1):"(IRR) ON AN INVESTMENT."
280 DISPLAY AT(5,1):"NEWTON'S METHOD IS USED"
290 DISPLAY AT(6,1):"TO TALLY THE IRR."
300 DISPLAY AT(8,1):"THIS REQUIRES AN INITIAL"
310 DISPLAY AT(9,1):"GUESS OF THE TRUE VALUE."
320 DISPLAY AT(11,1):"DON'T WORRY, I GUESS FOR"
330 DISPLAY AT(12,1):"YOU."
340 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
350 ACCEPT AT(23,25):Z$
360 RETURN
370 REM ENTER DATA
380 REM NUMBER OF PERIODS
390 GOSUB 460
400 REM CASH FLOW
410 REM INITIAL PERIOD
420 GOSUB 540
430 REM OTHER PERIODS
440 GOSUB 650
450 RETURN
```

```
460 REM NUMBER OF PERIODS
470 CALL CLEAR
480 DISPLAY AT(1,1):"HOW MANY PERIODS ARE IN"
490 DISPLAY AT(2,1):"YOUR CASH FLOW ?"
500 ACCEPT AT(2,18)BEEP VALIDATE(DIGIT,""):N$
510 IF N$="" THEN 500 ELSE N=VAL(N$)
520 IF N=0 THEN DISPLAY AT(23,1):"AT LEAST 1 PERIO
    D IS NEEDED" :: GOTO 500
530 RETURN
540 REM INITIAL PERIOD
550 CALL CLEAR
560 DISPLAY AT(1,1):"PLEASE ENTER THE INITIAL"
570 DISPLAY AT(2,1):"COST OF YOUR INVESTMENT."
580 DISPLAY AT(4,1):"FOR EXAMPLE, IF A NEW"
590 DISPLAY AT(5,1):"MACHINE COSTS $10,000 ,"
600 DISPLAY AT(6,1):"ENTER 10000."
610 DISPLAY AT(8,1):"COST = ?"
620 ACCEPT AT(8,10)BEEP VALIDATE(NUMERIC,""):V$
630 IF V$="" THEN 620 ELSE R(0)=-VAL(V$)
640 RETURN
650 REM OTHER PERIODS
660 CALL CLEAR
670 DISPLAY AT(1,1):"PLEASE ENTER THE NET REVENUE"
680 DISPLAY AT(2,1):"EXPECTED IN EACH PERIOD OF"
690 DISPLAY AT(3,1):"YOUR CASH FLOW."
700 DISPLAY AT(6,1):"'NET' REVENUE IS TOTAL"
710 DISPLAY AT(7,1):"REVENUE MINUS TOTAL COST."
720 FOR I=1 TO N
730 DISPLAY AT(11,1):"PERIOD #";I;TAB(13);"= ?"
740 ACCEPT AT(11,17)VALIDATE(NUMERIC,""):V$
750 IF V$="" THEN 740 ELSE R(I)=VAL(V$)
760 NEXT I
770 RETURN
780 REM COMPUTE
790 CALL CLEAR
800 DISPLAY AT(12,1):"COMPUTING ..."
810 REM TALLY COEFFICIENTS OF POLYNOMIAL & ITS DER
    IVATIVE
820 GOSUB 910
830 REM INITIAL GUESS OF IRR IN DECIMAL FORM
840 DATA 0.2
850 REM DELTA FOR SUCCESSIVE GUESSES
860 DATA 0.1
870 READ GUESS,DELTA
880 REM COMPUTE
890 GOSUB 1010
900 RETURN
910 REM COEFFICIENTS
920 REM POLYNOMIAL
```

```
930 FOR I=Ø TO N
940 POLY(I)=R(N-I)
950 NEXT I
960 REM DERIVATIVE
970 FOR I=Ø TO N
980 DERV(I)=I*POLY(I)
990 NEXT I
1000 RETURN
1010 REM COMPUTE
1020 REM UP TO 10 INITIAL GUESSES
1030 FOR I=1 TO 10
1040 DISPLAY AT(14,1):"CURRENT GUESS =";GUESS*100;
     "%"
1050 F$="NO GO"
1060 REM ESTIMATE ITERATIVELY
1070 GOSUB 1140
1080 REM CHECK FOR CONVERGENCE & FOR A NON-NEGATIV
     E IRR
1090 IF F$="GO!" AND RATE>=Ø THEN I=10 :: GOTO 112
     Ø
1100 REM TRY AGAIN
1110 GUESS=GUESS+DELTA
1120 NEXT I
1130 RETURN
1140 REM ESTIMATE
1150 REM NEW & OLD ESTIMATES OF IRR
1160 RATE=GUESS
1170 OLDE=-999
1180 REM MAKE UP TO 50 ESTIMATES
1190 FOR Q=1 TO 50
1200 REM EVALUATE POLYNOMIAL & DERIVATIVE AT PRESE
     NT ESTIMATE
1210 GOSUB 1320
1220 REM MAKE A NEW 'INITIAL' GUESS IF DERIVATIVE
     IS Ø
1230 IF D=Ø THEN Q=50 :: GOTO 1300
1240 REM NEW ESTIMATE
1250 RATE=RATE-P/D
1260 REM CHECK FOR CONVERGENCE
1270 IF RATE>=Ø AND ABS(RATE-OLDE)<Ø.Ø1 THEN F$="G
     O!" :: Q=50
1280 REM MAKE OLD ESTIMATE EQUAL TO NEW
1290 OLDE=RATE
1300 NEXT Q
1310 RETURN
1320 REM EVALUATE
1330 P=Ø
1340 FOR L=Ø TO N
1350 P=P+POLY(L)*(1+RATE)^L
```

```
1360 NEXT L
1370 D=0
1380 FOR L=1 TO N
1390 D=D+DERV(L)*(1+RATE)^(L-1)
1400 NEXT L
1410 RETURN
1420 REM CONVERGENCE
1430 CALL CLEAR
1440 CALL HCHAR(5,3,61,28)
1450 DISPLAY AT(6,6):"INVESTMENT RESULTS"
1460 CALL HCHAR(7,3,61,28)
1470 IMAGE IS ####.## %.
1480 DISPLAY AT(10,1):"THE INTERNAL RATE OF RETURN
     "
1490 DISPLAY AT(11,1):"ON YOUR INVESTMENT"
1500 DISPLAY AT(12,1):USING 1470:RATE*100
1510 CALL HCHAR(23,3,61,28)
1520 RETURN
1530 REM NON-CONVERGENCE
1540 CALL CLEAR
1550 DISPLAY AT(5,1):"SORRY, ESTIMATES OF THE IRR"
1560 DISPLAY AT(6,1):"ARE NOT CONVERGING."
1570 DISPLAY AT(8,1):"TRY DOUBLING THE VALUE OF"
1580 DISPLAY AT(9,1):"THE INITIAL GUESS"
1590 DISPLAY AT(10,1):"(LINE 840)."
1600 DISPLAY AT(12,1):"IF THIS DOESN'T WORK, DOUBL
     E"
1610 DISPLAY AT(13,1):"IT AGAIN, AND SO ON."
1620 RETURN
```

Least-Squares Forecasting

For centuries mankind has tried to peer into the future. Soothsayers, oracles, bone-throwers, and economists have all tried to tell us what the future holds—and now you can join this elite group of mystics by using the TI Least-Squares Forecasting program.

Suppose you want to predict what interest rates will be in 1985. You hypothesize that they will depend upon the rate of inflation. If inflation spurts ahead, you expect interest rates to rise; if inflation slows, you believe the rates will fall.

Table 2-1 lists data concerning changes in the consumer price index (inflation) and Treasury bill rates (interest). Enter this data into the forecasting routine, with interest rate as the dependent variable (Y) and inflation rate as the explanatory variable (X). The TI then calculates the A and B values for

your regression equation. In this case, that equation becomes:

Predicted Y = 1.807 + 0.749 * X

The computer also calculates R-squared, or the coefficient of determination. R-squared is the proportion of variation in the dependent variable explained by the regression equation. This statistic always ranges from 0 to 1, with a value close to unity suggesting that X explains Y pretty well.

Table 2-1. Data for Least-Squares Forecasting

Year	Three-Month Treasury Bill Rate, %	Annual Percent Change in the CPI, %
1968	5.3	4.2
1969	6.7	5.4
1970	6.4	5.9
1971	4.3	4.2
1972	4.1	3.3
1973	7.0	6.2
1974	7.9	11.0
1975	5.8	9.2
1976	5.0	5.7
1977	5.3	6.5
1978	7.2	7.6
1979	10.0	11.3
1980	11.6	13.5
1981	14.1	10.3
1982	10.7	6.2

To forecast the value of interest rates in 1985, enter the rate of inflation that you expect to prevail in that year. If you choose 7 percent per annum, the TI produces a predicted Y value of 7.168 and gives you 2.512 and 11.825 as the upper and lower bounds of the 95 percent confidence interval. Thus, your forecast is that interest rates in 1985 will equal 7.168 percent. The 95 percent confidence interval means that you are 95 percent sure that the actual interest rate will be between 2.512 percent and 11.825 percent. Note, however, that this range is computed under the strict assumption that the value of X, or the rate of inflation in 1985, is known with perfect certainty.

Finally, is your interest rate forecast any better than the one Madame Zelna makes by reading palms? Only time will tell.

Program 2-3. Least-Squares Forecasting

```
100 REM LEAST SQUARES FORECASTING
130 REM INITIALIZE
140 GOSUB 260
150 REM ENTER DATA
160 GOSUB 680
170 REM EDIT DATA
180 GOSUB 970
190 REM COMPUTE
200 GOSUB 1320
210 REM DISPLAY RESULTS
220 GOSUB 1960
230 REM FORECAST
240 GOSUB 2130
250 END
260 REM INITIALIZE
270 REM TITLE
280 GOSUB 340
290 REM HEADING
300 GOSUB 400
310 REM NUMBER OF OBSERVATIONS
320 GOSUB 550
330 RETURN
340 REM TITLE
350 CALL CLEAR
360 DISPLAY AT(12,9):"LEAST-SQUARES"
370 DISPLAY AT(14,10):"FORECASTING"
380 FOR DELAY=1 TO 500 :: NEXT DELAY
390 RETURN
400 REM HEADING
410 CALL CLEAR
420 DISPLAY AT(1,1):"THIS PROGRAM ESTIMATES A"
430 DISPLAY AT(2,1):"SIMPLE LINEAR REGRESSION"
440 DISPLAY AT(3,1):"EQUATION."
450 DISPLAY AT(5,1):"FUTURE VALUES OF THE"
460 DISPLAY AT(6,1):"DEPENDENT VARIABLE (Y) ARE"
470 DISPLAY AT(7,1):"PREDICTED, BASED ON THE"
480 DISPLAY AT(8,1):"VALUE OF X THAT YOU ENTER."
490 DISPLAY AT(10,1):"A 95% CONFIDENCE INTERVAL"
500 DISPLAY AT(11,1):"IS GENERATED FOR THE"
510 DISPLAY AT(12,1):"FORECAST."
520 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
530 ACCEPT AT(24,25):Z$
```

```
540 RETURN
550 REM NUMBER OF OBSERVATIONS
560 DATA 150
570 DIM X(150,2)
580 READ MAXN
590 CALL CLEAR
600 DISPLAY AT(1,1):"THE MAXIMUM NUMBER OF"
610 DISPLAY AT(2,1):"OBSERVATIONS ALLOWED"
620 DISPLAY AT(3,1):"IS";MAXN;"."
630 DISPLAY AT(5,1):"CHANGE LINES 560 & 570"
640 DISPLAY AT(6,1):"FOR A DIFFERENT LIMIT."
650 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
660 ACCEPT AT(24,25):Z$
670 RETURN
680 REM ENTER DATA
690 REM ON Y
700 GOSUB 740
710 REM ON X
720 GOSUB 870
730 RETURN
740 REM DATA ON Y
750 CALL CLEAR
760 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
770 DISPLAY AT(2,1):"DEPENDENT VARIABLE (Y)."
780 DISPLAY AT(4,1):"HIT 'ENTER' WHEN THROUGH."
790 N=MAXN
800 FOR I=1 TO MAXN
810 DISPLAY AT(8,1):"Y(";I;TAB(8);")= ?"
820 ACCEPT AT(8,13)VALIDATE(NUMERIC,""):S$
830 IF S$="" THEN N=I-1 :: I=MAXN ELSE X(I,1)=VAL(
    S$)
840 NEXT I
850 IF N<3 THEN DISPLAY AT(23,1):"AT LEAST 3 OBSER
    VATIONS ARE" :: DISPLAY AT(24,1):"NEEDED. PLEA
    SE TRY AGAIN." :: GOTO 800
860 RETURN
870 REM DATA ON X
880 CALL CLEAR
890 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
900 DISPLAY AT(2,1):"EXPLANATORY VARIABLE (X)."
910 FOR I=1 TO N
920 DISPLAY AT(7,1):"X(";I;TAB(8);")= ?"
930 ACCEPT AT(7,13)VALIDATE(NUMERIC,""):S$
940 IF S$="" THEN X(I,2)=0 ELSE X(I,2)=VAL(S$)
950 NEXT I
960 RETURN
970 REM EDIT DATA
980 V$(1)="Y" :: V$(2)="X"
990 FOR I=1 TO 2
```

```
1000 FOR L=0 TO INT((N-1)/10)
1010 REM DISPLAY DATA
1020 GOSUB 1080
1030 REM CORRECT DATA
1040 GOSUB 1160
1050 NEXT L
1060 NEXT I
1070 RETURN
1080 REM DISPLAY DATA
1090 CALL CLEAR
1100 DISPLAY AT(1,1):"THESE ARE VALUES OF ";V$(I);
     ":"
1110 FOR J=1 TO 10
1120 Q=J+L*10
1130 IF Q<=N THEN DISPLAY AT(J+2,1):V$(I);"(";Q;TA
     B(8);")= ";X(Q,I)
1140 NEXT J
1150 RETURN
1160 REM CORRECT DATA
1170 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1180 ACCEPT AT(16,20)VALIDATE("YN","")BEEP SIZE(1)
     :S$
1190 IF S$="" THEN 1180
1200 IF S$="N" THEN 1310
1210 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1220 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
1230 ACCEPT AT(19,24)BEEP VALIDATE(DIGIT,""):Q$
1240 IF Q$="" THEN 1230 ELSE Q=VAL(Q$)
1250 IF Q<(1+L*10)OR Q>N OR Q>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER NUMBER WITHIN" :: DI
     SPLAY AT(22,1):"BOUNDS SHOWN." :: GOTO 1230
1260 DISPLAY AT(21,1):"WHAT SHOULD THE"
1270 DISPLAY AT(22,1):"VALUE BE ?"
1280 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1290 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
1300 GOSUB 1090 :: GOTO 1170
1310 RETURN
1320 REM COMPUTE
1330 CALL CLEAR
1340 DISPLAY AT(12,1):"COMPUTING ..."
1350 REM REGRESSION EQUATION
1360 GOSUB 1420
1370 REM STANDARD ERRORS
1380 GOSUB 1630
1390 REM T-STATISTICS
1400 GOSUB 1740
1410 RETURN
1420 REM EQUATION
1430 REM KEY SUMS
```

```
1440 SX=Ø :: SY=Ø :: XQ=Ø :: YQ=Ø :: CP=Ø
1450 FOR I=1 TO N
1460 SX=SX+X(I,2)
1470 SY=SY+X(I,1)
1480 XQ=XQ+X(I,2)^2
1490 YQ=YQ+X(I,1)^2
1500 CP=CP+X(I,1)*X(I,2)
1510 NEXT I
1520 REM A & B
1530 B=(N*CP-SX*SY)/(N*XQ-SX*SX)
1540 A=(SY-B*SX)/N
1550 REM TERMS FOR FORECAST VARIANCE
1560 REM MEAN OF X
1570 SUM=Ø
1580 FOR I=1 TO N :: SUM=SUM+X(I,2):: NEXT I
1590 MEAN=SUM/N
1600 REM SUM OF SQUARED DEVIATIONS
1610 SD=XQ-SX*SX/N
1620 RETURN
1630 REM STANDARD ERRORS
1640 REM ANOVA TERMS
1650 TSS=YQ-SY*SY/N
1660 RSS=B*(CP-SX*SY/N)
1670 ESS=TSS-RSS
1680 V=N-2
1690 EV=ESS/V
1700 REM STANDARD ERRORS OF A & B
1710 SB=SQR(EV/(XQ-SX*SX/N))
1720 SA=SQR(EV*XQ/(N*XQ-SX*SX))
1730 RETURN
1740 REM T-VALUES
1750 REM VALUES FOR APPROXIMATION FORMULA
1760 DATA 1.96,Ø.6ØØ33,Ø.9591,-Ø.9Ø259,Ø.11588
1770 READ A1,B1,C1,D1,E1
1780 REM COMPUTE -- ACTUAL VALUES FOR LESS THAN 4
     D.O.F.
1790 IF V<4 THEN GOSUB 1810 ELSE GOSUB 1880
1800 RETURN
1810 REM ACTUAL VALUES
1820 DATA 12.706,4.303,3.182
1830 FOR I=1 TO 3
1840 READ VALUE
1850 IF V=I THEN T=VALUE
1860 NEXT I
1870 RETURN
1880 REM APPROXIMATION
1890 REM NUMERATOR
1900 NU=A1*V+B1+C1/V
1910 REM DENOMINATOR
```

```
1920 DE=V+D1+E1/V
1930 REM QUOTIENT
1940 T=NU/DE
1950 RETURN
1960 REM DISPLAY RESULTS
1970 CALL CLEAR
1980 CALL HCHAR(1,3,61,28)
1990 DISPLAY AT(2,6):"REGRESSION EQUATION"
2000 CALL HCHAR(3,3,61,28)
2010 DISPLAY AT(5,8):"ESTIMATED";TAB(26);"T"
2020 DISPLAY AT(6,1):"TERM";TAB(10);"VALUE";TAB(24
     );"RATIO"
2030 IMAGE ##########.### #######.###
2040 IMAGE = #.###
2050 DISPLAY AT(8,2):"A" :: DISPLAY AT(8,3):USING
     2030:A,A/SA
2060 DISPLAY AT(9,2):"B" :: DISPLAY AT(9,3):USING
     2030:B,B/SB
2070 DISPLAY AT(12,1):"R-SQUARED"
2080 DISPLAY AT(12,11):USING 2040:RSS/TSS
2090 CALL HCHAR(23,3,61,28)
2100 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
2110 ACCEPT AT(24,26):Z$
2120 RETURN
2130 REM FORECAST
2140 CALL CLEAR
2150 CALL HCHAR(1,3,61,28)
2160 DISPLAY AT(2,10):"FORECASTS"
2170 CALL HCHAR(3,3,61,28)
2180 DISPLAY AT(5,1):"WOULD YOU LIKE TO FORECAST"
2190 DISPLAY AT(6,1):"THE VALUE OF Y (Y/N) ?"
2200 ACCEPT AT(6,24)BEEP VALIDATE("YN","")SIZE(1):
     S$
2210 IF S$="" THEN 2200
2220 IF S$="N" THEN 2430
2230 DISPLAY AT(8,1):"WHAT IS THE VALUE"
2240 DISPLAY AT(9,1):"OF X ?"
2250 ACCEPT AT(9,8)BEEP VALIDATE(NUMERIC,""):S$
2260 IF S$="" THEN 2250 ELSE VX=VAL(S$)
2270 IMAGE = ########.###
2280 DISPLAY AT(12,1):"PREDICTED Y"
2290 P=A+B*VX
2300 DISPLAY AT(12,14):USING 2270:P
2310 DISPLAY AT(14,1):"95% CONFIDENCE"
2320 DISPLAY AT(15,1):"INTERVAL:"
2330 REM FORECAST VARIANCE
2340 FV=EV*(1+1/N+(VX-MEAN)^2/SD)
2350 DISPLAY AT(17,2):"LOWER BOUND"
2360 DISPLAY AT(17,14):USING 2270:P-T*SQR(FV)
```

```
237Ø DISPLAY AT(18,2):"UPPER BOUND"
238Ø DISPLAY AT(18,14):USING 227Ø:P+T*SQR(FV)
239Ø CALL HCHAR(23,3,61,28)
24ØØ DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
241Ø ACCEPT AT(24,26):Z$
242Ø GOTO 214Ø
243Ø RETURN
```

Time-Series Forecasting

The problem with least-squares forecasting is that future values of one variable are needed before future values of another can be predicted. In the preceding example, for instance, an estimate of the rate of inflation in 1985 was required to forecast the value of interest rates in that same year.

Time-series forecasting skirts this problem by making predictions of the future value of a variable, using past observations on that same term. A time series is a set of observations on a variable, in chronological order, at a set frequency. Monthly rainfall, yearly income, and daily stock-market prices are examples.

The time-series forecasting routine lets you use any of the following extrapolation techniques to predict the future value of a variable:

- Least-Squares Trend
- Semi-Averages
- Percent Change
- First Difference
- Past Averages

Using these techniques, you can make predictions as far into the future as you dare to go.

In the least-squares trend routine, your TI forecasts by extrapolating a historically fitted regression line into the future. For example, suppose you want to predict the value of the gross national product through 1984, using the historical data shown in Table 2-2. After you type the observations into the computer, the TI fits a least-squares line to the data. Observations over time are then generated by the TI, with 1972 corresponding to 1, 1973 corresponding to 2, and so on. Forecasts for 1983 and 1984 (time periods 12 and 13) are $3209 and $3405, respectively. Remember, these figures represent billions of dollars.

In the semi-averages routine, the time series is divided into two roughly equal parts. Means are computed for each part, and a straight line is then fitted through the two points, with values on the line representing forecasts. This method yields a 1983 GNP estimate of $3208 and a 1984 GNP estimate of $3404.

The percent-change routine calculates the percentage difference between the last two values in the time series. In this case, it is 4.12 percent. This percentage is then applied to future years. Hence, predicted GNP for 1983 is 1.0412 x $3059 = $3185. Using the same approach, the predicted GNP for 1984 is 1.0412 x $3185 = $3316.

The first-difference routine works much like the percent-change routine, except that it uses the difference (instead of percent change) between the last two values of the time series to forecast future values. For the example given here, that difference is $3059 − 2938 = $121. Hence, GNP forecasts for 1983 and 1984 are $3180 (or $3059 + $121) and $3301 (or $3180 + $121) respectively.

Finally, the past-average routine computes percent changes and first differences using an average of any number of past computations, not just one. For example, the mean of the past two first differences is ($2938 − $2633) + ($3059 − $2938) all divided by 2, or $213. Hence, predicted GNP in 1983 is $3059 + $213 = $3272.

Each of the preceding extrapolation techniques is based on the adage that "the past is prologue." If it isn't, perhaps we ought to read tea leaves instead.

Table 2-2. Gross National Product (In Billions of Dollars)

Year	GNP
1972	$1186
1973	1326
1974	1434
1975	1549
1976	1718
1977	1918
1978	2164
1979	2418
1980	2633
1981	2938
1982	3059

Program 2-4. Time-Series Forecasting

```
100 REM TIME-SERIES FORECASTING
130 REM INITIALIZE
140 GOSUB 260
150 REM ENTER DATA
160 GOSUB 640
170 REM EDIT DATA
180 GOSUB 770
190 REM CHOOSE METHOD
200 GOSUB 1100
210 IF CHOICE=6 THEN 250
220 REM MAKE PROJECTION
230 GOSUB 1240
240 GOTO 200
250 END
260 REM INITIALIZE
270 REM TITLE
280 GOSUB 340
290 REM HEADING
300 GOSUB 400
310 REM CHOICES
320 GOSUB 570
330 RETURN
340 REM TITLE
350 CALL CLEAR
360 DISPLAY AT(12,9):"TIME-SERIES"
370 DISPLAY AT(14,11):"FORECASTING"
380 FOR DELAY=1 TO 500 :: NEXT DELAY
390 RETURN
400 REM HEADING
410 CALL CLEAR
420 DATA 150
430 DIM Y(150)
440 READ MAXN
450 DISPLAY AT(1,1):"THIS PROGRAM FORECASTS"
460 DISPLAY AT(2,1):"FUTURE VALUES OF A TIME"
470 DISPLAY AT(3,1):"SERIES USING A HOST OF"
480 DISPLAY AT(4,1):"TREND ANALYSIS TECHNIQUES."
490 DISPLAY AT(6,1):"THE MAXIMUM NUMBER OF"
500 DISPLAY AT(7,1):"OBSERVATIONS ALLOWED"
510 DISPLAY AT(8,1):"IS";MAXN;"."
520 DISPLAY AT(10,1):"CHANGE LINES 420 & 430"
530 DISPLAY AT(11,1):"FOR A DIFFERENT LIMIT."
540 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
550 ACCEPT AT(24,25):Z$
560 RETURN
570 REM CHOICES
580 DATA LEAST-SQUARES TREND,SEMI-AVERAGES,PERCENT
     CHANGE
```

```
590 DATA FIRST DIFFERENCE,PAST AVERAGE,NONE
600 FOR I=1 TO 6
610 READ C$(I)
620 NEXT I
630 RETURN
640 REM ENTER DATA
650 CALL CLEAR
660 DISPLAY AT(1,1):"PLEASE ENTER OBSERVATIONS"
670 DISPLAY AT(2,1):"ON YOUR TIME SERIES. HIT"
680 DISPLAY AT(3,1):"'ENTER' WHEN THROUGH."
690 N=MAXN
700 FOR I=1 TO MAXN
710 DISPLAY AT(6,1):"PERIOD #(";I;TAB(14);")="
720 ACCEPT AT(6,17)VALIDATE(NUMERIC,""):S$
730 IF S$="" THEN N=I-1 :: I=MAXN ELSE Y(I)=VAL(S$
    )
740 NEXT I
750 IF N<3 THEN DISPLAY AT(23,1):"SORRY, AT LEAST
    3" :: DISPLAY AT(24,1):"OBSERVATIONS ARE NEEDE
    D" :: GOTO 690
760 RETURN
770 REM EDIT DATA
780 FOR L=0 TO INT((N-1)/10)
790 REM DISPLAY DATA
800 GOSUB 850
810 REM CORRECT DATA
820 GOSUB 940
830 NEXT L
840 RETURN
850 REM DISPLAY DATA
860 CALL CLEAR
870 DISPLAY AT(1,1):"THESE ARE VALUES OF YOUR"
880 DISPLAY AT(2,1):"TIME SERIES:"
890 FOR J=1 TO 10
900 M=J+L*10
910 IF M<=N THEN DISPLAY AT(J+3,1):"PERIOD (";M;TA
    B(13);")=";Y(M)
920 NEXT J
930 RETURN
940 REM CORRECT DATA
950 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
960 ACCEPT AT(16,20)VALIDATE("YN","")SIZE(1)BEEP:S
    $
970 IF S$="" THEN 960
980 IF S$="N" THEN 1090
990 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1000 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
1010 ACCEPT AT(19,24)BEEP VALIDATE(DIGIT,""):Q$
1020 IF Q$="" THEN 1010 ELSE Q=VAL(Q$)
```

```
1030 IF Q<(1+L*10)OR Q>N OR Q>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER NUMBER" :: DISPLAY A
     T(22,1):"WITHIN BOUNDS SHOWN." :: GOTO 1010
1040 DISPLAY AT(21,1):"WHAT SHOULD THE"
1050 DISPLAY AT(22,1):"VALUE BE ?"
1060 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1070 IF S$="" THEN Y(Q)=0 ELSE Y(Q)=VAL(S$)
1080 GOSUB 850 :: GOTO 950
1090 RETURN
1100 REM CHOOSE METHOD
1110 CALL CLEAR
1120 DISPLAY AT(1,1):"FUTURE VALUES OF YOUR TIME"
1130 DISPLAY AT(2,1):"SERIES ARE PROJECTED USING"
1140 DISPLAY AT(3,1):"ANY OF THESE METHODS:"
1150 ROW=6
1160 FOR I=1 TO 6
1170 DISPLAY AT(ROW,2):STR$(I)&". ";C$(I)
1180 ROW=ROW+2
1190 NEXT I
1200 DISPLAY AT(22,1):"YOUR CHOICE = ?"
1210 ACCEPT AT(22,17)VALIDATE("123456","")SIZE(1)B
     EEP:S$
1220 IF S$="" THEN 1210 ELSE CHOICE=VAL(S$)
1230 RETURN
1240 REM MAKE PROJECTIONS
1250 REM NUMBER OF FUTURE PERIODS
1260 GOSUB 1340
1270 REM PROJECTIONS
1280 CALL CLEAR
1290 IF CHOICE<>5 THEN DISPLAY AT(12,1):"FORECASTI
     NG ..."
1300 ON CHOICE GOSUB 1480,1660,1920,1970,2020
1310 REM DISPLAY
1320 GOSUB 2710
1330 RETURN
1340 REM NUMBER OF FUTURE PERIODS
1350 CALL CLEAR
1360 DISPLAY AT(1,1):"THE LAST PERIOD OF YOUR TIME
     "
1370 DISPLAY AT(2,1):"SERIES IS NUMBER";N;"."
1380 DISPLAY AT(5,1):"HOW MANY PERIODS INTO THE"
1390 DISPLAY AT(6,1):"FUTURE DO YOU WANT TO"
1400 DISPLAY AT(7,1):"FORECAST ?"
1410 ACCEPT AT(7,12)BEEP VALIDATE(DIGIT,""):S$
1420 IF S$="" THEN 1410 ELSE NF=VAL(S$)
1430 IF NF=0 THEN 1410
1440 REM CHECK FOR MEMORY CONSTRAINT
1450 T=N+NF
```

```
1460 IF T>MAXN THEN DISPLAY AT(22,1):"SORRY, ONLY"
     ;MAXN-N;"MORE PERIODS" :: DISPLAY AT(23,1):"A
     RE ALLOWED." :: GOTO 1410
1470 RETURN
1480 REM LEAST-SQUARES TREND
1490 REM KEY SUMS
1500 SX=0 :: SY=0 :: XQ=0 :: YQ=0 :: CP=0
1510 FOR I=1 TO N
1520 SX=SX+I
1530 SY=SY+Y(I)
1540 XQ=XQ+I*I
1550 YQ=YQ+Y(I)^2
1560 CP=CP+Y(I)*I
1570 NEXT I
1580 REM A&B
1590 B=(N*CP-SX*SY)/(N*XQ-SX*SX)
1600 A=(SY-B*SX)/N
1610 REM FORECASTS
1620 FOR I=N+1 TO T
1630 Y(I)=A+B*I
1640 NEXT I
1650 RETURN
1660 REM METHOD OF SEMI-AVERAGES
1670 REM NUMBER OF POINTS IN EACH GROUP
1680 G1=INT(N/2)
1690 G2=N-G1
1700 REM GROUP MEANS
1710 Y1=0 :: X1=0
1720 FOR I=1 TO G1
1730 Y1=Y1+Y(I)
1740 X1=X1+I
1750 NEXT I
1760 Y1=Y1/G1 :: X1=X1/G1
1770 REM MEANS OF SECOND GROUP
1780 Y2=0 :: X2=0
1790 FOR I=G1+1 TO N
1800 Y2=Y2+Y(I)
1810 X2=X2+I
1820 NEXT I
1830 Y2=Y2/G2 :: X2=X2/G2
1840 B=(Y2-Y1)/(X2-X1)
1850 REM Y-INTERCEPT
1860 A=Y2-B*X2
1870 REM FORECASTS
1880 FOR I=N+1 TO T
1890 Y(I)=A+B*I
1900 NEXT I
1910 RETURN
1920 REM PERCENT CHANGE
```

```
1930 FOR I=N+1 TO T
1940 Y(I)=Y(I-1)*Y(I-1)/Y(I-2)
1950 NEXT I
1960 RETURN
1970 REM FIRST DIFFERENCE
1980 FOR I=N+1 TO T
1990 Y(I)=Y(I-1)+(Y(I-1)-Y(I-2))
2000 NEXT I
2010 RETURN
2020 REM PAST AVERAGE
2030 DISPLAY AT(1,1):"WOULD YOU LIKE TO USE AN"
2040 DISPLAY AT(2,1):"AVERAGE OF PAST:"
2050 DISPLAY AT(5,3):"1. % CHANGES"
2060 DISPLAY AT(7,3):"2. FIRST DIFFERENCES"
2070 DISPLAY AT(9,3):"3. ACTUAL VALUES"
2080 DISPLAY AT(12,1):"CHOICE = ?"
2090 ACCEPT AT(12,12)BEEP VALIDATE("123","")SIZE(1
     ):S$
2100 IF S$="" THEN 2090 ELSE AVG=VAL(S$)
2110 IF AVG=1 THEN T$="PERCENT CHANGES"
2120 IF AVG=2 THEN T$="FIRST DIFFERENCES"
2130 ON AVG GOSUB 2180,2180,2270
2140 CALL CLEAR
2150 DISPLAY AT(12,1):"FORECASTING ..."
2160 ON AVG GOSUB 2350,2470,2590
2170 RETURN
2180 REM % CHANGES OR 1ST DIFFERENCES
2190 DISPLAY AT(15,1):"HOW MANY PAST"
2200 DISPLAY AT(16,1):T$;" DO YOU"
2210 DISPLAY AT(17,1):"WANT TO USE ?"
2220 ACCEPT AT(17,15)BEEP VALIDATE(DIGIT,""):P$
2230 IF P$="" THEN 2220 ELSE PP=VAL(P$)
2240 IF PP=0 THEN 2220
2250 IF PP+1>N THEN DISPLAY AT(23,1):"SORRY, ONLY"
     ;N-1;"AVAILABLE" :: GOTO 2220
2260 RETURN
2270 REM ACTUAL VALUES
2280 DISPLAY AT(15,1):"HOW MANY PAST ACTUAL VALUES
     "
2290 DISPLAY AT(16,1):"WOULD YOU LIKE TO USE ?"
2300 ACCEPT AT(16,24)BEEP VALIDATE(DIGIT,""):P$
2310 IF P$="" THEN 2300 ELSE PP=VAL(P$)
2320 IF PP=0 THEN 2300
2330 IF PP>N THEN DISPLAY AT(23,1):"SORRY, ONLY";N
     ;"AVAILABLE" :: GOTO 2300
2340 RETURN
2350 REM PERCENT CHANGES
2360 REM PAST AVERAGE
2370 PC=0
```

```
2380 FOR I=N TO N-PP+1 STEP -1
2390 PC=PC+Y(I)/Y(I-1)
2400 NEXT I
2410 PC=PC/PP
2420 REM FORECASTS
2430 FOR I=N+1 TO T
2440 Y(I)=Y(I-1)*PC
2450 NEXT I
2460 RETURN
2470 REM FIRST DIFFERENCES
2480 REM PAST AVERAGE
2490 FD=0
2500 FOR I=N TO N-PP+1 STEP -1
2510 FD=FD+Y(I)-Y(I-1)
2520 NEXT I
2530 FD=FD/PP
2540 REM FORECASTS
2550 FOR I=N+1 TO T
2560 Y(I)=Y(I-1)+FD
2570 NEXT I
2580 RETURN
2590 REM ACTUAL VALUES
2600 REM PAST AVERAGE
2610 AV=0
2620 FOR I=N TO N-PP+1 STEP -1
2630 AV=AV+Y(I)
2640 NEXT I
2650 AV=AV/PP
2660 REM FORECASTS
2670 FOR I=N+1 TO T
2680 Y(I)=AV
2690 NEXT I
2700 RETURN
2710 REM DISPLAY
2720 FOR L=0 TO INT((NF-1)/10)
2730 REM HEADING
2740 GOSUB 2790
2750 REM BODY
2760 GOSUB 2890
2770 NEXT L
2780 RETURN
2790 REM HEADING
2800 CALL CLEAR
2810 CALL HCHAR(1,3,61,28)
2820 DISPLAY AT(2,11):"FORECASTS"
2830 CALL HCHAR(3,3,61,28)
2840 DISPLAY AT(4,1):"Method: ";C$(CHOICE)
```

```
285Ø IF CHOICE=5 AND AVG=1 THEN DISPLAY AT(4,22):"
     OF";PP :: DISPLAY AT(5,9):"% CHANGES"
286Ø IF CHOICE=5 AND AVG=2 THEN DISPLAY AT(4,22):"
     OF";PP :: DISPLAY AT(5,9):"FIRST DIFFERENCES"
287Ø IF CHOICE=5 AND AVG=3 THEN DISPLAY AT(4,22):"
     OF";PP :: DISPLAY AT(5,9):"ACTUAL VALUES"
288Ø RETURN
289Ø REM BODY
29ØØ IMAGE ##########.###
291Ø FOR J=1 TO 1Ø
292Ø M=J+L*1Ø+N
293Ø IF M<=T THEN DISPLAY AT(J+7,1):"Period (";M;T
     AB(13);")=" :: DISPLAY AT(J+7,15): USING 29ØØ
     :Y(M)
294Ø NEXT J
295Ø CALL HCHAR(23,3,61,28)
296Ø DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
297Ø ACCEPT AT(24,26):Z$
298Ø RETURN
```

Computer Cash Register

This program turns your TI into a fast-working and easy-to-use cash register. You enter the price of an item. The TI responds with the total payment due from the buyer, including tax. After you enter the amount of money received from the customer, the TI tells you how much change to give. It's as simple as that. You can redo a messed-up transaction by pressing R, and you can view total sales at any time by pressing T.

Figure 2-5. Computer Cash Register

```
**********************************************************************
                    $$ CASH REGISTER $$
**********************************************************************
                      PRICE  =  $  99.95
                   WITH TAX  =  $ 109.95
                    PAYMENT  =  $ 120.00
                     CHANGE  =  $  10.05
**********************************************************************
                     PRESS        TO:
                       C          CONTINUE
                       R          REDO LAST ENTRY
                       T          TALLY TOTALS
**********************************************************************
```

Program 2-5. Computer Cash Register

```
100 REM COMPUTER CASH REGISTER
130 REM INITIALIZE
140 GOSUB 200
150 REM DRAW BOX
160 GOSUB 330
170 REM OPERATE
180 GOSUB 510
190 END
200 REM INITIALIZE
210 CALL CLEAR
220 DISPLAY AT(1,1):"THIS PROGRAM TURNS YOUR TI"
230 DISPLAY AT(2,1):"INTO A CASH REGISTER."
240 DISPLAY AT(4,1):"PLEASE ENTER THE PERCENTAGE"
250 DISPLAY AT(5,1):"SALES TAX IN YOUR AREA."
260 DISPLAY AT(7,1):"FOR EXAMPLE, ENTER 5 FOR 5%,"
270 DISPLAY AT(8,1):"7 FOR 7%, AND SO ON."
280 DISPLAY AT(10,1):"TAX % = ?"
290 ACCEPT AT(10,11)BEEP VALIDATE(NUMERIC,""):T$
300 IF T$="" THEN 290 ELSE TRATE=1+VAL(T$)/100
310 SALES=0 :: N=0 :: TAX=0
320 RETURN
330 REM DRAW BOX
340 CALL CLEAR
350 CALL HCHAR(1,3,42,28)
360 CALL HCHAR(3,3,42,28)
370 CALL HCHAR(17,3,42,28)
380 CALL HCHAR(24,3,42,28)
390 CALL VCHAR(1,3,42,24)
400 CALL VCHAR(1,31,42,24)
410 DISPLAY AT(2,6):"$$ CASH REGISTER $$"
420 DISPLAY AT(18,3):"PRESS";TAB(17);"TO:"
430 DISPLAY AT(20,5):"C";TAB(12);"CONTINUE"
440 DISPLAY AT(21,5):"R";TAB(12);"REDO LAST ENTRY"
450 DISPLAY AT(22,5):"T";TAB(12);"TALLY TOTALS"
460 DISPLAY AT(7,6):"Price = $"
470 DISPLAY AT(8,3):"With Tax = $"
480 DISPLAY AT(11,4):"Payment = $"
490 DISPLAY AT(13,5):"Change = $"
500 RETURN
510 REM OPERATE
520 REM ENTER PRICE & PAYMENT
530 GOSUB 680
540 REM ENTER NEXT ACTION
550 GOSUB 860
560 REM RESPOND
570 REM REDO
580 IF CHR$(R)="R" THEN DISPLAY AT(5,3):"Let's try
     that again!" :: GOTO 530
```

```
590 DISPLAY AT(5,3):""
600 REM ADD TO TOTALS
610 GOSUB 990
620 REM CONTINUE
630 IF CHR$(R)="C" THEN GOTO 530
640 REM DISPLAY TOTALS
650 IF CHR$(R)="T" THEN GOSUB 1040
660 IF S$="Y" THEN GOSUB 330 :: GOTO 530
670 RETURN
680 REM ENTER FIGURES
690 REM BLANK-OUT AREAS
700 DISPLAY AT(7,16):""
710 DISPLAY AT(8,16):""
720 DISPLAY AT(11,16):""
730 DISPLAY AT(13,16):""
740 REM PRICE
750 ACCEPT AT(7,16)BEEP VALIDATE(NUMERIC,""):S$
760 IF S$="" THEN 750 ELSE PRICE=VAL(S$)
770 PWT=INT((PRICE*TRATE+0.005)*100)/100
780 DISPLAY AT(8,15):PWT
790 REM PAYMENT
800 ACCEPT AT(11,16)BEEP VALIDATE(NUMERIC,""):S$
810 IF S$="" THEN 800 ELSE PAYMENT=VAL(S$)
820 IF PAYMENT>=PWT THEN 840
830 DISPLAY AT(11,16):"I WANT MORE!" :: FOR DELAY=
    1 TO 300 :: NEXT DELAY :: GOTO 800
840 DISPLAY AT(13,15):PAYMENT-PWT
850 RETURN
860 REM NEXT ACTION
870 FOR DELAY=1 TO 3
880 DISPLAY AT(18,3)SIZE(5):""
890 CALL KEY(0,R,S)
900 IF S<>0 THEN 970
910 NEXT DELAY
920 FOR DELAY=1 TO 3
930 DISPLAY AT(18,3)SIZE(5):"PRESS"
940 CALL KEY(0,R,S)
950 IF S<>0 THEN 970
960 NEXT DELAY
970 IF R<>67 AND R<>82 AND R<>84 THEN 870
980 RETURN
990 REM ADD TO TOTALS
1000 SALES=SALES+PRICE
1010 TAXES=TAXES+(PWT-PRICE)
1020 N=N+1
1030 RETURN
1040 REM DISPLAY TOTALS
1050 CALL CLEAR
1060 CALL HCHAR(1,3,61,28)
```

```
1070 DISPLAY AT(2,12):"TOTALS"
1080 CALL HCHAR(3,3,61,28)
1090 IMAGE = $########.##
1100 IMAGE =  ########.##
1110 DISPLAY AT(5,1):"TOTAL SALES"
1120 DISPLAY AT(5,13):USING 1090:SALES
1130 DISPLAY AT(7,1):"TOTAL TAXES"
1140 DISPLAY AT(7,13):USING 1090:TAXES
1150 DISPLAY AT(9,9):"SUM"
1160 DISPLAY AT(9,13):USING 1090:SALES+TAXES
1170 DISPLAY AT(12,3):"NUMBER OF" :: DISPLAY AT(13
     ,7):"SALES"
1180 DISPLAY AT(13,13):USING 1100:N
1190 CALL HCHAR(20,3,61,28)
1200 DISPLAY AT(21,1):"WOULD YOU LIKE TO KEEP YOUR
     "
1210 DISPLAY AT(22,1):"CASH REGISTER ON (Y/N) ?_"
1220 CALL KEY(0,K,ST)
1230 IF ST=0 THEN 1220
1240 IF K<>78 AND K<>89 THEN 1220
1250 S$=CHR$(K)
1260 RETURN
```

Chapter 3

Games

Games

This chapter presents the following games, each designed to test your knowledge, concentration, persistence, and intelligence:

- "Hide Brer Rabbit." Try to guess a word in order to prevent Brer Fox and Brer Bear from finding Brer Rabbit, who is hiding in the briar patch. If you're playing alone, the computer will select a word for you. This game is for two or more players; try selecting words for each other.
- "Rings and Poles." Your job? To move a number of different-sized rings from Pole 1 to Pole 2 in as few moves as possible. There's only one catch—you can't place a larger ring on top of a smaller one. This game is also called Towers of Hanoi.
- "Matches." It sounds simple enough: You want to pick up the last match in a row of matches. Alternate moves with another player or with the computer. The computer graciously lets you go first—but it might not do you any good.
- "Vanilla Cookie." Try to make your opponent eat the vanilla part of the cookie. Two to five players are allowed. This game, also called "Chomp," is from an old *Scientific American* article and is a real brainteaser.

Brer Rabbit

Brer Rabbit is in the briar patch, playing hide-and-seek with Brer Fox and Brer Bear. Your job is to keep him hidden by guessing a word. If you're playing alone, the computer will randomly select a word for you from its dictionary of 104 choices. If you are playing against another person, try selecting words for each other. Words can be from 3 to 15 characters long.

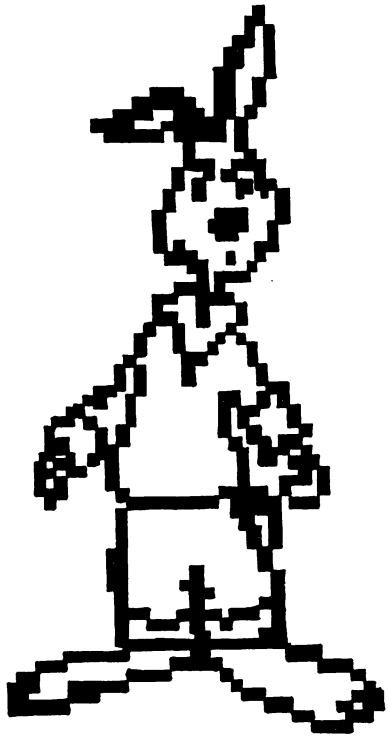
To play this game, simply enter letters that you think might be in the word. If you're right, the computer pinpoints the position of the letter. But if you're wrong, the computer starts to reveal Brer Rabbit. First come the ears, followed by the head, body, arms, legs, feet, and tail—seven parts in all. If you don't guess the word in time, Brer Rabbit (Figure 3-1) is discovered.

If your TI always selects the same word when you turn on your machine, try entering RANDOMIZE followed by a number before running the program. Be sure to use a different number each time you power up. Finally, when you think you've learned all of the computer's vocabulary, you may

want to rejuvenate it by adding words of your own (lines 2600–2860).

One hint: Try entering vowels, particularly *i* and *e*, first. The pattern of vowels may help you figure out the word.

Figure 3-1. Brer Rabbit



Program 3-1. Hide Brer Rabbit

```
100 REM HIDE BRER RABBIT
110 REM
120 REM
130 REM INITIALIZE
140 GOSUB 250
150 REM PLAY GAME
160 GOSUB 480
170 REM PLAY AGAIN
180 DISPLAY AT(23,1):"Would you like to play"
190 DISPLAY AT(24,1):"again (Y/N) ?"
```

```
200 CALL KEY(0,K,S)
210 IF S=0 THEN 200
220 IF CHR$(K)="Y" THEN 160
230 IF CHR$(K)<>"N" THEN 200
240 END
250 REM INITIALIZE
260 CALL CLEAR
270 DISPLAY AT(1,1):"Welcome to 'Guess a Word,'"
280 DISPLAY AT(2,1):"or 'Hide Brer Rabbit.'"
290 DISPLAY AT(4,1):"Brer Rabbit is in the briar"
300 DISPLAY AT(5,1):"patch, playing hide-and-seek"
310 DISPLAY AT(6,1):"with Brer Fox and Brer Bear."
320 DISPLAY AT(8,1):"Your job is to keep him"
330 DISPLAY AT(9,1):"hidden by guessing a word."
340 DISPLAY AT(11,1):"If you're playing alone,"
350 DISPLAY AT(12,1):"I'll select the word for"
360 DISPLAY AT(13,1):"you."
370 DISPLAY AT(15,1):"If two of you are playing,"
380 DISPLAY AT(16,1):"try selecting words for each
    "
390 DISPLAY AT(17,1):"other."
400 DISPLAY AT(19,1):"Up to 7 wrong letters are"
410 DISPLAY AT(20,1):"allowed before Brer Rabbit"
420 DISPLAY AT(21,1):"is revealed."
430 DISPLAY AT(24,1):"Press any key to continue"
440 CALL KEY(0,K,S)
450 IF S=0 THEN 440
460 RANDOMIZE
470 RETURN
480 REM PLAY GAME
490 REM SELECT WORD
500 CALL CLEAR
510 DISPLAY AT(1,1):"Would you like me to select"
520 DISPLAY AT(2,1)BEEP:"the word (Y/N) ?"
530 CALL KEY(0,K,S)
540 IF S=0 THEN 530
550 IF CHR$(K)<>"Y" AND CHR$(K)<>"N" THEN 530
560 IF CHR$(K)="Y" THEN GOSUB 2540 ELSE GOSUB 600
570 REM GAME
580 GOSUB 710
590 RETURN
600 REM PLAYER SELECTS
610 CALL CLEAR
620 DISPLAY AT(1,1):"Please enter the word that"
630 DISPLAY AT(2,1):"you want your opponent to"
640 DISPLAY AT(3,1):"guess.  Up to 15 characters"
650 DISPLAY AT(4,1):"are allowed."
660 DISPLAY AT(7,1):"Word ?"
670 ACCEPT AT(7,8)BEEP VALIDATE(UALPHA,"")SIZE(15)
    :WORD$
```

```
680 IF WORD$="" THEN 670
690 IF LEN(WORD$)<3 THEN DISPLAY AT(23,1):"That's
    too easy! Try again." :: GOTO
670
700 RETURN
710 REM GAME
720 REM INITIAL POSITION
730 GOSUB 830
740 REM ENTER LETTER
750 GOSUB 990
760 REM ENTER GUESS
770 IF H$=WORD$ THEN GUESS$=WORD$ ELSE GOSUB 2210
780 REM CHECK FOR WINNER
790 IF GUESS$=WORD$ THEN GOSUB 2380 :: GOTO 820
800 REM CHECK FOR LOSER
810 IF N=7 THEN GOSUB 2460 ELSE GOTO 750
820 RETURN
830 REM INITIAL POSITION
840 N=0 :: GUESS$=""
850 L=LEN(WORD$)
860 LETTERS$=""
870 REM HIDDEN WORD
880 H$=RPT$("-",L)
890 CALL CLEAR
900 CALL HCHAR(1,3,61,28)
910 DISPLAY AT(2,7):"HIDE BRER RABBIT"
920 CALL HCHAR(3,3,61,28)
930 DISPLAY AT(14,1):"No. of Bad Letters =";N
940 DISPLAY AT(16,1):"Bad Letters"
950 DISPLAY AT(17,1):"Used: "
960 DISPLAY AT(19,1):"Word: ";H$
970 CALL HCHAR(24,3,61,28)
980 RETURN
990 REM LETTER
1000 DISPLAY AT(21,1):""
1010 DISPLAY AT(22,1):""
1020 DISPLAY AT(21,7)BEEP:"Press any letter"
1030 CALL KEY(0,K,S)
1040 IF S=0 THEN 1030
1050 IF K<65 OR K>90 THEN 1030
1060 REM MAKE SURE LETTER IS NOT ALREADY IN WORD
1070 GOSUB 1230
1080 IF CK$="" THEN 1110
1090 DISPLAY AT(21,1)BEEP:"Letter is already in wo
     rd !"
1100 FOR DELAY=1 TO 1000 :: NEXT DELAY :: GOTO 100
     0
1110 REM SEARCH FOR LETTER
1120 F$="A MISS"
```

```
1130 GOSUB 1290
1140 REM ADD TO BODY
1150 IF F$="A MISS" THEN GOSUB 1180 :: GOSUB 1360
1160 DISPLAY AT(19,7):H$
1170 RETURN
1180 REM BAD LETTERS
1190 LETTERS$=LETTERS$&CHR$(K)&","
1200 DISPLAY AT(17,7):LETTERS$
1210 CALL SOUND(1000,110,0)
1220 RETURN
1230 REM CHECK FOR LETTER ALREADY IN WORD
1240 CK$=""
1250 FOR I=1 TO L
1260 IF SEG$(H$,I,1)=CHR$(K)THEN CK$="USED"
1270 NEXT I
1280 RETURN
1290 REM SEARCH
1300 HD$=""
1310 FOR I=1 TO L
1320 IF SEG$(WORD$,I,1)=CHR$(K)THEN HD$=HD$&CHR$(K
     ):: F$="HIT" ELSE HD$=HD$&SEG$(H$,I,1)
1330 NEXT I
1340 H$=HD$
1350 RETURN
1360 REM ADD TO BODY
1370 N=N+1
1380 DISPLAY AT(14,21):N
1390 DISPLAY AT(2,1):""
1400 ON N GOSUB 1420,1530,1660,1770,1880,1990,2160
1410 RETURN
1420 REM EARS
1430 DISPLAY AT(2,6):"The ears are shown"
1440 CALL CHAR(126,"0000000206050D19")
1450 CALL CHAR(127,"0000000000000301")
1460 CALL CHAR(128,"0000000E3FF3C7FB")
1470 CALL CHAR(129,"1931323232B4E8E8")
1480 DISPLAY AT(4,14):CHR$(126)
1490 DISPLAY AT(5,12):CHR$(127)
1500 DISPLAY AT(5,13):CHR$(128)
1510 DISPLAY AT(5,14):CHR$(129)
1520 RETURN
1530 REM HEAD
1540 DISPLAY AT(2,6):"The head is shown"
1550 CALL CHAR(130,"0101010202040408")
1560 CALL CHAR(131,"0804CE528F880038")
1570 CALL CHAR(132,"0000000000808080")
1580 CALL CHAR(133,"08080B0701000B0C")
1590 CALL CHAR(134,"7931031284B8A0F0")
1600 DISPLAY AT(6,13):CHR$(130)
```

```
1610 DISPLAY AT(6,14):CHR$(131)
1620 DISPLAY AT(6,15):CHR$(132)
1630 DISPLAY AT(7,13):CHR$(133)
1640 DISPLAY AT(7,14):CHR$(134)
1650 RETURN
1660 REM BODY
1670 DISPLAY AT(2,6):"The body is shown"
1680 CALL CHAR(135,"080C12222241A1A1")
1690 CALL CHAR(136,"8888102844840202")
1700 CALL CHAR(137,"2040404080800000")
1710 CALL CHAR(138,"01323C3824120B09")
1720 DISPLAY AT(8,13):CHR$(135)
1730 DISPLAY AT(8,14):CHR$(136)
1740 DISPLAY AT(9,13):CHR$(137)
1750 DISPLAY AT(9,14):CHR$(138)
1760 RETURN
1770 REM ARMS
1780 DISPLAY AT(2,6):"The arms are shown"
1790 CALL CHAR(139,"03061834426253B5")
1800 CALL CHAR(140,"8080404020C0A050")
1810 CALL CHAR(141,"C9A1E04000000000")
1820 CALL CHAR(142,"28C8887000000000")
1830 DISPLAY AT(9,12)SIZE(1):CHR$(139)
1840 DISPLAY AT(9,15):CHR$(140)
1850 DISPLAY AT(10,12)SIZE(1):CHR$(141)
1860 DISPLAY AT(10,15):CHR$(142)
1870 RETURN
1880 REM LEGS
1890 DISPLAY AT(2,6):"The legs are shown"
1900 CALL CHAR(143,"0000807F80808080")
1910 CALL CHAR(34,"080830D010080404")
1920 CALL CHAR(35,"8080818381818E39D")
1930 CALL CHAR(36,"0202010181183DD21")
1940 DISPLAY AT(10,13)SIZE(1):CHR$(143)
1950 DISPLAY AT(10,14)SIZE(1):CHR$(34)
1960 DISPLAY AT(11,13):CHR$(35)
1970 DISPLAY AT(11,14):CHR$(36)
1980 RETURN
1990 REM FEET
2000 DISPLAY AT(2,6):"The feet are shown"
2010 CALL CHAR(37,"0000000003050607")
2020 CALL CHAR(38,"00001FE000031CE0")
2030 CALL CHAR(60,"817F810778800000")
2040 CALL CHAR(62,"827F80402010C03")
2050 CALL CHAR(42,"00807E01000184")
2060 CALL CHAR(43,"00000000008040C0")
2070 CALL CHAR(91,"631E000000000000")
2080 DISPLAY AT(12,11):CHR$(37)
2090 DISPLAY AT(12,12):CHR$(38)
```

```
2100 DISPLAY AT(12,13):CHR$(60)
2110 DISPLAY AT(12,14):CHR$(62)
2120 DISPLAY AT(12,15):CHR$(42)
2130 DISPLAY AT(12,16):CHR$(43)
2140 DISPLAY AT(13,15):CHR$(91)
2150 RETURN
2160 REM TAIL
2170 DISPLAY AT(2,6):"The tail is shown"
2180 CALL CHAR(92,"0808 3EDFDE0905 05")
2190 DISPLAY AT(10,14)SIZE(1):CHR$(92)
2200 RETURN
2210 REM GUESS
2220 DISPLAY AT(21,1):"Would you like to guess the
     "
2230 DISPLAY AT(22,1)BEEP:"word (Y/N) ?"
2240 CALL KEY(0,K,S)
2250 IF S=0 THEN 2240
2260 IF CHR$(K)<>"Y" AND CHR$(K)<>"N" THEN 2240
2270 IF CHR$(K)="N" THEN 2370
2280 DISPLAY AT(21,1):"What is your"
2290 DISPLAY AT(22,1):"guess ?"
2300 ACCEPT AT(22,9)BEEP SIZE(15)VALIDATE(UALPHA,"
     "):GUESS$
2310 IF GUESS$="" THEN 2300
2320 IF GUESS$=WORD$ THEN GOTO 2370
2330 DISPLAY AT(21,1):""
2340 DISPLAY AT(21,11):"WRONG !"
2350 DISPLAY AT(22,1):""
2360 FOR DELAY=1 TO 10 :: CALL SOUND(100,110,1)::
     NEXT DELAY
2370 RETURN
2380 REM WINNER
2390 CALL CLEAR
2400 DISPLAY AT(1,7):"CONGRATULATIONS !"
2410 DISPLAY AT(3,1):"You correctly guessed the"
2420 DISPLAY AT(4,1):"word ";WORD$;"."
2430 DISPLAY AT(7,1):"You did this using";N
2440 DISPLAY AT(8,1):"bad letters."
2450 RETURN
2460 REM LOSER
2470 CALL CLEAR
2480 DISPLAY AT(1,7):"MY CONDOLENCES !"
2490 DISPLAY AT(3,1):"Brer Fox and Brer Bear have"
2500 DISPLAY AT(4,1):"found Brer Rabbit."
2510 DISPLAY AT(6,1):"The correct word was:"
2520 DISPLAY AT(8,9):WORD$
2530 RETURN
2540 REM COMPUTER SELECTS
2550 RESTORE
```

```
2560 C=INT(RND*104)+1
2570 FOR I=1 TO C
2580 READ WORD$
2590 NEXT I
2600 DATA ABLE,APPLE,AIRPLANE,ARTILLERYMAN
2610 DATA BOAT,BELLYBUTTON,BAT,BENEDICTION
2620 DATA CAT,CATERPILLAR,COATS,CATHOLIC
2630 DATA CORN,COSMOPOLITAN,COUNT,COUNTRY
2640 DATA DOG,DRAG,DECEIT,DECOY
2650 DATA EEL,EFFICIENCY,EGGHEAD,ELECTRICIAN
2660 DATA FATHER,FEATHER,FELONY,FINICKY
2670 DATA GOAT,GREAT,GLADIATOR,GNOME
2680 DATA HIPPOPOTAMUS,HOLOCAUST,HOLY,HONEY
2690 DATA IDEAL,IMPRISON, ICEMAN, IMMEASURABLE
2700 DATA JOWL,JAW,JUGGLER,JUVENILE
2710 DATA KALEIDOSCOPE,KEEL,KANGAROO,KING
2720 DATA LABOR,LADLE,LANGUAGE,LISTEN
2730 DATA MALLARD,MAP,MANSLAUGHTER,MERCENARY
2740 DATA NIGHT,NICKEL,NUCLEAR,NUMBER
2750 DATA OIL,OPPORTUNITY,OUTSTANDING,OZONE
2760 DATA PACK,PAINTER,PENGUIN,POLYNOMIAL
2770 DATA QUACK,QUAGMIRE,QUARTER,QUEASY
2780 DATA RUFFLE,RANDOMIZE,REFINED,RHEUMATIC
2790 DATA STASH,SNIFFLE,SWEEPSTAKES,SYNAGOGUE
2800 DATA TATTOO,TEMPESTUOUS,THIRSTY,TUNA
2810 DATA UNCOUTH,UNCLE,UPON,USHER
2820 DATA VERB,VAT,VICTORY,VOLUNTEER
2830 DATA WALRUS,WART,WHITTLE,WORKABLE
2840 DATA XYLOPHONE,IMAGINATION,ILLUMINATION,HARMO
     NICA
2850 DATA YAM,YEOMANLY,YOGURT,YOUTHFUL
2860 DATA ZEBRA,ZESTFUL,ZODIAC,ZOOLOGICAL
2870 RETURN
```

Rings and Poles

Also called Towers of Hanoi, this game is guaranteed to stimulate your gray matter. You are presented with three poles and from three to nine rings. Each ring is of a different size. You start with all rings on Pole 1, smallest on top and largest on bottom, as Figure 3-2 shows.

Your objective is to transfer the rings from Pole 1 to Pole 2, one at a time, in as few moves as possible. There's a catch: You can't place a larger ring on top of a smaller one. This means that you'll have to use Pole 3 for temporary storage, placing rings on it until they're needed elsewhere.

Your best strategy is to begin with simple cases. Figure 3-3 gives a step-by-step account of a three-ring game; it requires seven moves.

Figure 3-2. Rings and Poles

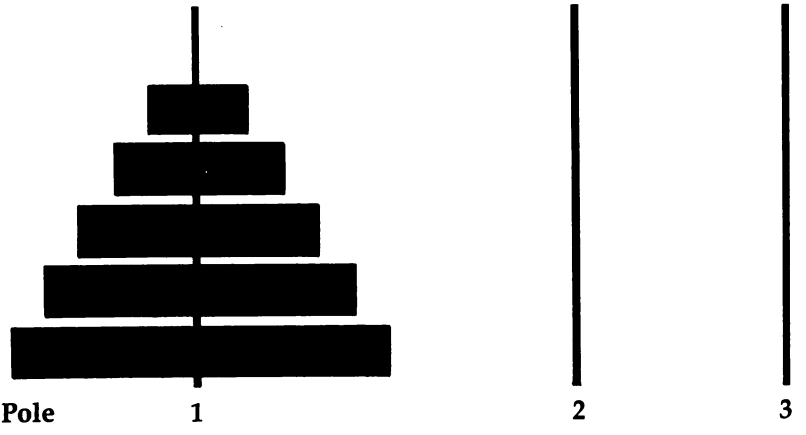
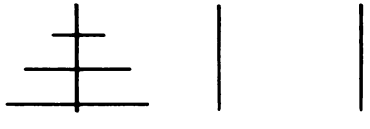
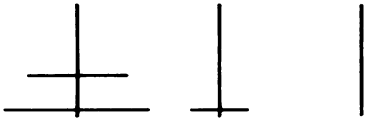


Figure 3-3. The Three-Ring Game

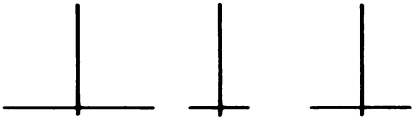
Initial Position



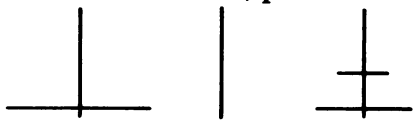
1. Take from Pole 1; place on Pole 2



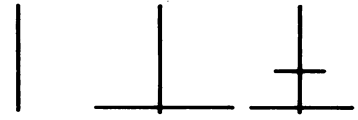
2. Take from Pole 1; place on Pole 3



3. Take from Pole 2; place on Pole 3

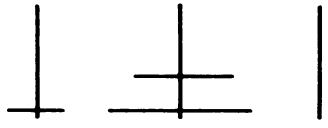


4. Take from Pole 1; place on Pole 2

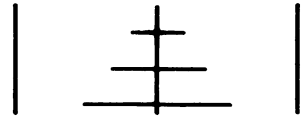


5. Take from Pole 3; place on Pole 1

6. Take from Pole 3; place on Pole 2



7. Take from Pole 1; place on Pole 2



Once you're comfortable with three rings, try it with five or six rings. You'll have to concentrate and think ahead to get a perfect score. Then when you feel up to it, try it with all nine rings—but you'd better save that one for a long weekend.

Program 3-2. Rings and Poles

```
100 REM POLES AND RINGS
130 REM INITIALIZE
140 GOSUB 250
150 REM PLAY GAME
160 GOSUB 740
170 REM CONTINUE
180 DISPLAY AT(23,4):"Would you like to play"
190 DISPLAY AT(24,4):"again (Y/N) ?"
200 CALL KEY(0,K,S)
210 IF S=0 THEN 200
220 IF CHR$(K)="Y" THEN 160
230 IF CHR$(K)<>"N" THEN 200
240 END
250 REM INITIALIZE
260 REM EXPLAIN RULES
270 GOSUB 350
280 REM CREATE RINGS & POLES
290 GOSUB 570
300 DIM TRK(3,10)
310 REM WARNING MESSAGES
320 W1$="You just took it from there!"
330 W2$="That's illegal !"
340 RETURN
350 REM RULES
360 CALL CLEAR
370 DISPLAY AT(1,1):"This is a game of Rings and"
380 DISPLAY AT(2,1):"Poles.  You have 3 Poles and"
390 DISPLAY AT(3,1):"3 to 9 Rings.  Each Ring is"
400 DISPLAY AT(4,1):"of a different size.  You"
410 DISPLAY AT(5,1):"start with all Rings on"
420 DISPLAY AT(6,1):"Pole 1, smallest on top, and"
430 DISPLAY AT(7,1):"largest on bottom."
440 DISPLAY AT(9,1):"Your objective is to"
450 DISPLAY AT(10,1):"transfer the Rings from"
460 DISPLAY AT(11,1):"Pole 1 to Pole 2, one at a"
470 DISPLAY AT(12,1):"time, in as few moves as"
480 DISPLAY AT(13,1):"possible."
490 DISPLAY AT(15,1):"Oh, there's a catch.  You"
500 DISPLAY AT(16,1):"can't place a larger Ring on
    "
510 DISPLAY AT(17,1):"top of a smaller one."
520 DISPLAY AT(19,1):"Good luck !"
530 DISPLAY AT(24,3):"Press any key to begin"
540 CALL KEY(0,K,S)
550 IF S=0 THEN 540
560 RETURN
570 REM CREATE RINGS & POLES
```

```
580 CALL CHAR(42,"FFFFFFFFFFFFFF00")
590 CALL CHAR(64,"FFFFFFFFFFFFFF18")
600 CALL CHAR(36,"0F0F0F0F0F0F0F00")
610 CALL CHAR(37,"F0F0F0F0F0F0F000")
620 CALL CHAR(38,"1818181818181818")
630 RG$(9)="****@****"
640 RG$(8)="$***@***%"
650 RG$(7)=" ***@*** "
660 RG$(6)=" $**@**% "
670 RG$(5)="  **@**  "
680 RG$(4)="  $*@*%  "
690 RG$(3)="{3 SPACES}*@*{3 SPACES}"
700 RG$(2)="{3 SPACES}$@%{3 SPACES}"
710 RG$(1)="{4 SPACES}@{4 SPACES}"
720 RG$(0)="{4 SPACES}&{4 SPACES}"
730 RETURN
740 REM PLAY GAME
750 REM ENTER NUMBER OF RINGS TO USE
760 GOSUB 840
770 REM DRAW INITIAL POSITION
780 GOSUB 920
790 REM TRACK INITIAL POSITION
800 GOSUB 1110
810 REM MAKE MOVES
820 GOSUB 1330
830 RETURN
840 REM NUMBER OF RINGS
850 CALL CLEAR
860 DISPLAY AT(1,1):"How many rings do you want"
870 DISPLAY AT(2,1):"to use ?"
880 ACCEPT AT(2,10)BEEP VALIDATE(DIGIT,"")SIZE(1):
    K$
890 IF K$="" THEN 880 ELSE K=VAL(K$)
900 IF K<3 THEN DISPLAY AT(23,1):"That's not very
    challenging!" :: GOTO 880
910 RETURN
920 REM INITIAL POSITION
930 REM MOVES
940 CALL CLEAR
950 DISPLAY AT(1,8):"RINGS AND POLES"
960 FINISH$="NO"
970 MOVE=0
980 DISPLAY AT(3,6):"Number of Moves =";MOVES
990 REM POLES
1000 CALL VCHAR(8,8,38,9)
1010 CALL VCHAR(8,17,38,9)
1020 CALL VCHAR(8,26,38,9)
1030 DISPLAY AT(19,1):"Pole 1";TAB(15);"2";TAB(24)
     ;"3"
```

```
1040 REM RINGS
1050 ROW=16
1060 FOR I=K TO 1 STEP -1
1070 DISPLAY AT(ROW,2)SIZE(9):RG$(I)
1080 ROW=ROW-1
1090 NEXT I
1100 RETURN
1110 REM SIZE OF THE RING AT EACH POSITION
1120 REM ALL POLES
1130 FOR I=1 TO 3
1140 FOR J=0 TO 10
1150 TRK(I,J)=0
1160 NEXT J
1170 NEXT I
1180 REM POLE 1
1190 SZ=K
1200 FOR I=9 TO 9-K+1 STEP -1
1210 TRK(1,I)=SZ
1220 SZ=SZ-1
1230 NEXT I
1240 REM LOCATION OF TOP RING ON EACH POLE
1250 LOC(1)=9-K+1
1260 LOC(2)=10
1270 LOC(3)=10
1280 REM SIZE OF EACH TOP RING
1290 SZE(1)=1
1300 SZE(2)=0
1310 SZE(3)=0
1320 RETURN
1330 REM MAKE MOVES
1340 REM ENTER NUMBER OF POLE TO TAKE A RING FROM
1350 MOVE$="OKAY"
1360 GOSUB 1450
1370 REM ENTER NUMBER OF POLE TO PUT IT ON
1380 GOSUB 1530
1390 IF MOVE$="TABOO" THEN GOTO 1350
1400 REM MAKE MOVE
1410 GOSUB 1630
1420 REM CHECK FOR FINISH
1430 IF TRK(2,9-K+1)=1 THEN GOSUB 1900 ELSE 1350
1440 RETURN
1450 REM TAKE
1460 DISPLAY AT(23,1):""
1470 DISPLAY AT(21,7):"Take From Pole #"
1480 ACCEPT AT(21,23)BEEP SIZE(1)VALIDATE("123",""
     ):T$
1490 IF T$="" THEN 1480 ELSE T=VAL(T$)
1500 REM CHECK FOR LEGAL MOVE
```

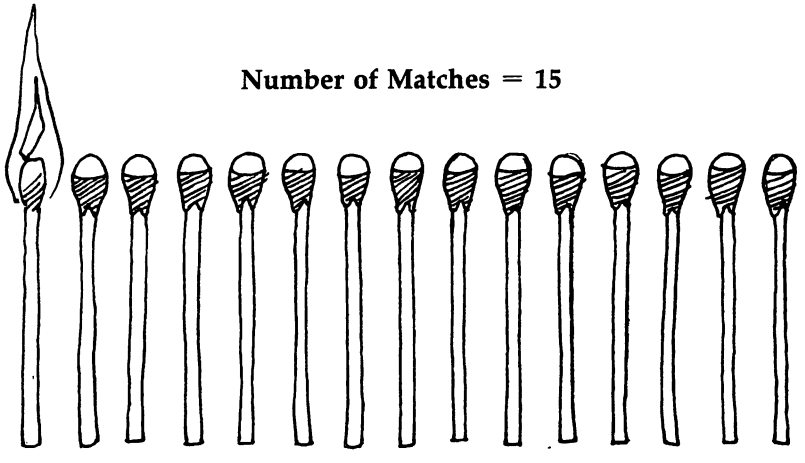
```
1510 IF SZE(T)=Ø THEN DISPLAY AT(23,1):"That would
     be a good trick!" :: GOTO 148Ø
1520 RETURN
1530 REM PLACE
1540 DISPLAY AT(23,1):""
1550 DISPLAY AT(21,7):"Place on Pole #"
1560 ACCEPT AT(21,22)BEEP VALIDATE("123","")SIZE(1
     ):P$
1570 IF P$="" THEN 156Ø ELSE P=VAL(P$)
1580 REM CHECK FOR SAME POLE
1590 IF P=T THEN CALL WARN(W1$,MOVE$):: GOTO 162Ø
1600 REM CHECK FOR LEGAL MOVE
1610 IF SZE(P)<>Ø AND SZE(T)>SZE(P)THEN CALL WARN(
     W2$,MOVE$)
1620 RETURN
1630 REM MAKE MOVE
1640 REM MAKE RING GO POOF
1650 GOSUB 171Ø
1660 REM MAKE RING APPEAR
1670 GOSUB 178Ø
1680 MOVE=MOVE+1
1690 DISPLAY AT(3,24):MOVE
1700 RETURN
1710 REM POOF
1720 IF T=1 THEN COL=2
1730 IF T=2 THEN COL=11
1740 IF T=3 THEN COL=2Ø
1750 DISPLAY AT(LOC(T)+7,COL)SIZE(9):RG$(Ø)
1760 TRK(T,LOC(T))=Ø
1770 RETURN
1780 REM MAKE RING RE-APPEAR
1790 IF P=1 THEN COL=2
1800 IF P=2 THEN COL=11
1810 IF P=3 THEN COL=2Ø
1820 DISPLAY AT(LOC(P)+6,COL)SIZE(9):RG$(SZE(T))
1830 LOC(P)=LOC(P)-1
1840 SZE(P)=SZE(T)
1850 LOC(T)=LOC(T)+1
1860 SZE(T)=TRK(T,LOC(T))
1870 REM TRACK
1880 TRK(P,LOC(P))=SZE(P)
1890 RETURN
1900 REM CHECK FOR FINISH
1910 CALL CLEAR
1920 CALL HCHAR(1,3,61,28)
1930 DISPLAY AT(2,7):"CONGRATULATIONS !"
1940 CALL HCHAR(3,3,61,28)
1950 REM TALLY PERFECT SCORE
1960 PS=3
```

```
1970 FOR I=3 TO K
1980 PS=2*PS+1
1990 NEXT I
2000 REM DISPLAY
2010 DISPLAY AT(7,2):"Perfect Score =";PS
2020 DISPLAY AT(8,5):"Your Score =";MOVE
2030 IF PS=MOVE THEN DISPLAY AT(11,2):"I'm impress
     ed !"
2040 CALL HCHAR(22,3,61,28)
2050 RETURN
2060 SUB WARN(F$,MOVE$)
2070 DISPLAY AT(23,1):F$
2080 FOR DELAY=1 TO 20
2090 CALL SOUND(100,300,0)
2100 NEXT DELAY
2110 MOVE$="TABOO"
2120 SUBEND
```

Matches

You're sitting at a table, facing a row of matches. Only the leftmost match is lit, however, and you want to be the first one to pick it up.

Figure 3-4. The Game of Matches

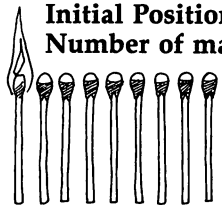


The row contains from 7 to 25 matches. Starting at the far right end of the row, you and your opponent take turns removing matches. You must take either 1, 2, or 3 at a time. You have the option of playing against the computer or against another person. But be forewarned: Although the computer always lets you go first, it also plays a perfect game.

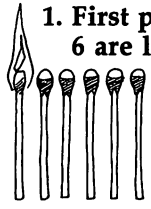
To learn the secret of Matches, pay careful attention to the computer's moves. Do you see a pattern emerging? You can truly buffalo your friends once you've learned the secret. A sample game between two players is shown in Figure 3-5.

Figure 3-5. A Sample Game of Matches

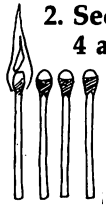
Initial Position
Number of matches = 9



1. First player takes 3 matches; 6 are left.



2. Second player takes 2 matches; 4 are left.



3. First player takes 1 match; 3 are left.

4. Second player takes 3 matches and wins.

Program 3-3. Matches

```
100 REM MATCHES
130 REM INITIALIZE
140 GOSUB 250
150 REM PLAY GAME
160 GOSUB 420
170 REM PLAY AGAIN
180 DISPLAY AT(23,1):"Would you like to play"
190 DISPLAY AT(24,1):"again (Y/N) ?"
200 CALL KEY(0,K,S)
210 IF S=0 THEN 200
220 IF CHR$(K)="Y" THEN 160
230 IF CHR$(K)<>"N" THEN 200
240 END
250 REM INITIALIZE
260 CALL CLEAR
270 DISPLAY AT(1,1):"This is the game of MATCHES."
280 DISPLAY AT(3,1):"I'll display between 7 and"
290 DISPLAY AT(4,1):"25 matches in a row."
300 DISPLAY AT(6,1):"Only the first one is lit,"
310 DISPLAY AT(7,1):"however, and your objective"
320 DISPLAY AT(8,1):"is to pick it up."
330 DISPLAY AT(10,1):"You and your opponent start"
340 DISPLAY AT(11,1):"at the end of the row and"
350 DISPLAY AT(12,1):"take away 1,2, or 3 sticks"
360 DISPLAY AT(13,1):"at a time."
370 DISPLAY AT(23,3):"Press any key to begin"
380 CALL KEY(0,K,S)
390 IF S=0 THEN 380
400 RANDOMIZE
410 RETURN
420 REM PLAY GAME
430 REM PLAYERS
440 GOSUB 540
450 REM INITIAL POSITION
460 GOSUB 830
470 REM MAKE MOVE
480 GOSUB 1130
490 REM END OF GAME
500 IF T=0 THEN GOSUB 1470 :: GOTO 530
510 IF PLAYER=2 THEN PLAYER=1 ELSE PLAYER=2
520 GOTO 480
530 RETURN
540 REM PLAYERS
550 REM SELECT OPPONENT
560 GOSUB 600
570 REM NAMES
580 GOSUB 710
```

```
590 RETURN
600 REM SELECT OPPONENT
610 CALL CLEAR
620 DISPLAY AT(1,1):"Two players are needed for"
630 DISPLAY AT(2,1):"MATCHES."
640 DISPLAY AT(4,1):"Would you like to play"
650 DISPLAY AT(5,1)BEEP:"against me (Y/N) ?"
660 CALL KEY(0,K,S)
670 IF S=0 THEN 660
680 IF CHR$(K)<>"Y" AND CHR$(K)<>"N" THEN 660
690 IF CHR$(K)="Y" THEN OPP$="COMPUTER" ELSE OPP$=
    "HUMAN"
700 RETURN
710 REM NAMES
720 CALL CLEAR
730 DISPLAY AT(1,1):"Please enter the name of"
740 DISPLAY AT(2,1):"each player."
750 REM FIRST
760 IF OPP$="COMPUTER" THEN K=1 :: NAME$(2)="COMPU
    TER" ELSE K=2
770 FOR I=1 TO K
780 DISPLAY AT(4,1):"Player No.";I;"="
790 ACCEPT AT(4,16)BEEP SIZE(12)VALIDATE(UALPHA,""
    ):NAME$(I)
800 IF NAME$(I)="" THEN 790
810 NEXT I
820 RETURN
830 REM INITIAL POSITION
840 REM NUMBER OF STICKS
850 T=7+INT(RND*19)
860 REM BOARD
870 GOSUB 910
880 REM COUNTER
890 PLAYER=1
900 RETURN
910 REM BOARD
920 REM CHARACTERS
930 GOSUB 970
940 REM DISPLAY
950 GOSUB 1030
960 RETURN
970 REM CHARACTERS
980 CALL COLOR(14,16,1)
990 CALL CHAR(136,"1818181818181818")
1000 CALL CHAR(128,"003C7E7E7E7E7E3C")
1010 CALL CHAR(129,"0010183424743810")
1020 RETURN
1030 REM DISPLAY
1040 CALL CLEAR
```

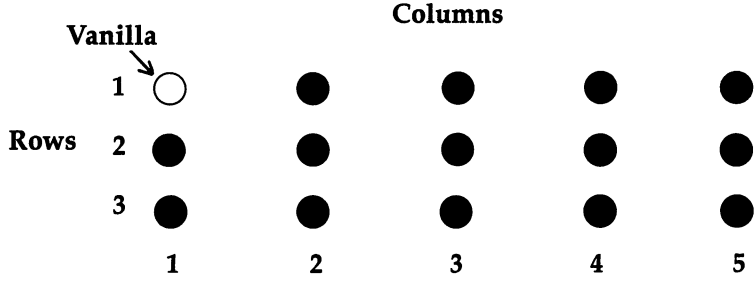
```
1050 DISPLAY AT(1,10):"MATCHES"
1060 DISPLAY AT(9,1):CHR$(129)
1070 FOR I=1 TO T
1080 DISPLAY AT(10,I):CHR$(128)
1090 DISPLAY AT(11,I):CHR$(136)
1100 DISPLAY AT(12,I):CHR$(136)
1110 NEXT I
1120 RETURN
1130 REM MAKE MOVE
1140 DISPLAY AT(6,1):"Matches left =";T
1150 DISPLAY AT(16,1):"Your turn ";NAME$(PLAYER);"
     ."
1160 IF OPP$="COMPUTER" AND NAME$(PLAYER)="COMPUTE
     R" THEN GOSUB 1200 ELSE GOSUB
1310
1170 REM TAKE STICKS AWAY
1180 GOSUB 1410
1190 RETURN
1200 REM COMPUTER
1210 REM NUMBER THAT SHOULD BE LEFT
1220 S=INT(T/4)*4
1230 REM NUMBER TO TAKE
1240 N=T-S
1250 IF N=0 THEN N=1
1260 DISPLAY AT(18,1):"Hmm ..."
1270 FOR DELAY=1 TO 1000 :: NEXT DELAY
1280 DISPLAY AT(18,1):"I'll take";N
1290 FOR DELAY=1 TO 1000 :: NEXT DELAY
1300 RETURN
1310 REM PLAYER
1320 DISPLAY AT(18,1):"How many do you want ?"
1330 ACCEPT AT(18,24)BEEP VALIDATE("123","")SIZE(1
     ):N$
1340 IF N$="" THEN 1330 ELSE N=VAL(N$)
1350 IF N<=T THEN 1400
1360 DISPLAY AT(23,1):"Too few left! Try again."
1370 FOR DELAY=1 TO 2 :: CALL SOUND(1000,110,0)::
     NEXT DELAY
1380 DISPLAY AT(23,1):""
1390 GOTO 1330
1400 RETURN
1410 REM TAKE STICKS AWAY
1420 DISPLAY AT(10,T-N+1):""
1430 DISPLAY AT(11,T-N+1):""
1440 DISPLAY AT(12,T-N+1):""
1450 T=T-N
1460 RETURN
1470 REM A WINNER
1480 CALL CLEAR
```

```
1490 DISPLAY AT(1,7):"CONGRATULATIONS !"
1500 DISPLAY AT(3,1):"You picked up the flaming"
1510 DISPLAY AT(4,1):"match, ";NAME$(PLAYER);", an
     d can"
1520 DISPLAY AT(5,1):"therefore light your victory
     "
1530 DISPLAY AT(6,1):"cake."
1540 IF PLAYER=2 THEN PLAYER=1 ELSE PLAYER=2
1550 DISPLAY AT(22,1):"So sorry, ";NAME$(PLAYER);"
     ."
1560 RETURN
```

Vanilla Cookie

After you've mastered Matches, you'll be ready for the more difficult game of Vanilla Cookie. It's designed for two to five players, who alternately take bites from a rectangular-shaped cookie (Figure 3-6). The objective is to force an opponent to consume the vanilla part—the white dot in the upper left-hand corner.

Figure 3-6. Vanilla Cookie



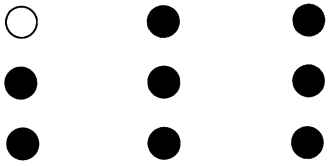
The size of the cookie can be set by you or by the TI. The smallest allowable size is two rows by two columns, while the largest possible cookie measures 10 rows by 13 columns.

Begin eating the cookie from the lower right-hand corner. Your bites can be any size you choose. The size and location of each bite are determined by row and column numbers that you enter at each turn. You'll eat all dots below and to the right of the point you designate, including that point. In a 10 x 10 cookie, for example, the entry ROW = 8 and COLUMN = 8

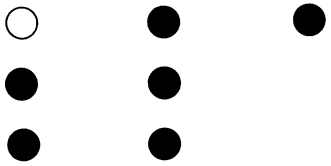
means that a 3 x 3 chunk is taken from the cookie's lower right-hand corner. A sample game is shown in Figure 3-7.

Figure 3-7. A Sample Two-Person Game

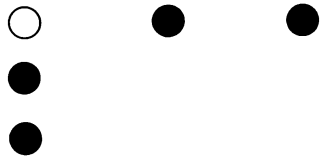
Initial Position



1. Player A enters:
Row = 2 and Col = 3



2. Player B responds:
Row = 2 and Col = 2



3. Player A enters:
Row = 2 and Col = 1

4. Player B responds:
Row = 1 and Col = 2

5. Player A is forced to eat the vanilla part of the cookie.

Originally called "Chomp," this game first appeared in the mathematical games section of *Scientific American* in the late 1960s. A winning strategy does exist for the two-player, square-cookie case, but for more general cases no strategy has been found—and that's one of the things that make this such a fascinating game.

Program 3-4. Vanilla Cookie (Chomp)

```
100 REM VANILLA COOKIE (CHOMP)
110 REM
120 REM
130 REM INITIALIZE
140 GOSUB 250
150 REM PLAY GAME
160 GOSUB 420
170 REM PLAY AGAIN
180 DISPLAY AT(23,1):"Would you like to play"
190 DISPLAY AT(24,1)BEEP:"again (Y/N) ?"
200 CALL KEY(0,K,S)
210 IF S=0 THEN 200
220 IF CHR$(K)="Y" THEN 160
230 IF CHR$(K)<>"N" THEN 200
240 END
250 REM INITIALIZE
260 CALL CLEAR
270 DISPLAY AT(1,1):"This is the game of Vanilla"
280 DISPLAY AT(2,1):"Cookie, or CHOMP."
290 DISPLAY AT(4,1):"The game is played by"
300 DISPLAY AT(5,1):"chomping away at a"
310 DISPLAY AT(6,1):"rectangular-shaped cookie."
320 DISPLAY AT(8,1):"The objective is to force"
330 DISPLAY AT(9,1):"your opponent to eat the"
340 DISPLAY AT(10,1):"vanilla part, shown in"
350 DISPLAY AT(11,1):"white."
360 DISPLAY AT(23,1):"Press any key to continue"
370 CALL KEY(0,K,S)
380 IF S=0 THEN 370
390 DIM NC(13)
400 RANDOMIZE
410 RETURN
420 REM PLAY GAME
430 REM PLAYERS
440 GOSUB 560
450 REM COOKIE SIZE
460 GOSUB 770
470 REM INITIAL POSITION
480 GOSUB 1130
```

```
490 REM BITE
500 GOSUB 1670
510 REM CHECK FOR END OF GAME
520 IF NC(1)=0 THEN GOSUB 2150 :: GOTO 550
530 IF PLAYER=N THEN PLAYER=1 ELSE PLAYER=PLAYER+1
540 GOTO 500
550 RETURN
560 REM PLAYERS
570 CALL CLEAR
580 DISPLAY AT(1,1):"How many players are"
590 DISPLAY AT(2,1):"there ?"
600 ACCEPT AT(2,9)BEEP VALIDATE(DIGIT,"")SIZE(1):N
    $
610 IF N$="" THEN 600 ELSE N=VAL(N$)
620 IF N<2 THEN DISPLAY AT(23,1):"SORRY, TWO PLAYE
    RS NEEDED" :: GOTO 600
630 IF N>5 THEN DISPLAY AT(23,1):"SORRY, ONLY 5 AL
    LOWED" :: GOTO 600
640 REM NAMES
650 GOSUB 670
660 RETURN
670 REM NAMES
680 CALL CLEAR
690 DISPLAY AT(1,1):"Please enter each player's"
700 DISPLAY AT(2,1):"name."
710 FOR I=1 TO N
720 DISPLAY AT(5,1):"Player";I;"="
730 ACCEPT AT(5,12)BEEP VALIDATE(UALPHA,"")SIZE(12
    ):NAME$(I)
740 IF NAME$(I)="" THEN 730
750 NEXT I
760 RETURN
770 REM COOKIE SIZE
780 CALL CLEAR
790 DISPLAY AT(1,1):"The maximum size of your"
800 DISPLAY AT(2,1):"cookie is 13 Rows by 10"
810 DISPLAY AT(3,1):"Columns."
820 DISPLAY AT(5,1):"The actual size can be"
830 DISPLAY AT(6,1):"generated randomly by me or"
840 DISPLAY AT(7,1):"deliberately by you."
850 DISPLAY AT(9,1):"Would you like me to do"
860 DISPLAY AT(10,1)BEEP:"it (Y/N) ?"
870 CALL KEY(0,K,S)
880 IF S=0 THEN 870
890 IF CHR$(K)<>"Y" AND CHR$(K)<>"N" THEN 870
900 IF CHR$(K)="Y" THEN GOSUB 920 ELSE GOSUB 960
910 RETURN
920 REM COMPUTER GENERATES SIZE
930 ROWS=INT(RND*12)+2
```

```
940 COLS=INT(RND*9)+2
950 RETURN
960 REM PLAYER GENERATES SIZE
970 REM ROWS
980 CALL CLEAR
990 DISPLAY AT(1,1):"Please enter the number of:"
1000 DISPLAY AT(4,2):"Rows ="
1010 ACCEPT AT(4,9)BEEP VALIDATE(DIGIT,""):R$
1020 IF R$="" THEN 1010 ELSE ROWS=VAL(R$)
1030 IF ROWS<2 THEN DISPLAY AT(23,1):"AT LEAST 2 N
     EEDED" :: GOTO 1010
1040 IF ROWS>13 THEN DISPLAY AT(23,1):"ONLY 13 ALL
     OWED" :: GOTO 1010
1050 REM COLUMNS
1060 DISPLAY AT(23,1):""
1070 DISPLAY AT(6,2):"Cols ="
1080 ACCEPT AT(6,9)BEEP VALIDATE(DIGIT,""):C$
1090 IF C$="" THEN 1080 ELSE COLS=VAL(C$)
1100 IF COLS<2 THEN DISPLAY AT(23,1):"AT LEAST 2 N
     EEDED" :: GOTO 1080
1110 IF COLS>10 THEN DISPLAY AT(23,1):"ONLY 10 ALL
     OWED" :: GOTO 1080
1120 RETURN
1130 REM INITIAL POSITION
1140 REM LABELING
1150 GOSUB 1210
1160 REM COOKIE
1170 GOSUB 1360
1180 REM COUNTERS
1190 GOSUB 1570
1200 RETURN
1210 REM LABELING
1220 CALL CLEAR
1230 DISPLAY AT(1,8):"VANILLA COOKIE"
1240 DISPLAY AT(3,12):"Columns"
1250 C=5
1260 FOR I=1 TO COLS
1270 DISPLAY AT(5,C):I
1280 C=C+2
1290 NEXT I
1300 DISPLAY AT(9,1):"R" :: DISPLAY AT(10,1):"o" :
     : DISPLAY AT(11,1):"w" :: DISPLAY AT(12,1):"s
     "
1310 FOR I=1 TO ROWS
1320 IF I<10 THEN V$=" "&STR$(I)ELSE V$=STR$(I)
1330 DISPLAY AT(I+6,2):V$
1340 NEXT I
1350 RETURN
1360 REM COOKIE
```

```
1370 A$="003C7E7E7E7E3C00"
1380 CALL CHAR(128,A$)
1390 CALL CHAR(136,A$)
1400 CALL COLOR(13,16,1)
1410 REM FIRST ROW
1420 DISPLAY AT(7,6):CHR$(128)
1430 C=8
1440 FOR I=2 TO COLS
1450 DISPLAY AT(7,C):CHR$(136)
1460 C=C+2
1470 NEXT I
1480 REM OTHER ROWS
1490 FOR I=2 TO ROWS
1500 C=6
1510 FOR J=1 TO COLS
1520 DISPLAY AT(I+6,C):CHR$(136)
1530 C=C+2
1540 NEXT J
1550 NEXT I
1560 RETURN
1570 REM COUNTERS
1580 REM PLAYER
1590 PLAYER=1
1600 REM NUMBER OF COLUMNS IN EACH ROW
1610 FOR I=1 TO 13
1620 IF I<=ROWS THEN NC(I)=COLS ELSE NC(I)=0
1630 NEXT I
1640 REM TOTAL NUMBER OF ROWS
1650 TR=ROWS
1660 RETURN
1670 REM BITE
1680 REM ENTER ROW
1690 GOSUB 1770
1700 REM ENTER COLUMN
1710 GOSUB 1870
1720 REM TAKE BITE
1730 GOSUB 1950
1740 REM UPDATE COUNTERS
1750 GOSUB 2090
1760 RETURN
1770 REM ENTER ROW
1780 DISPLAY AT(21,1):"Please bite ";NAME$(PLAYER)
     ;"."
1790 REM ROW
1800 DISPLAY AT(23,3):"Row ="
1810 ACCEPT AT(23,9)BEEP SIZE(2)VALIDATE(DIGIT,"")
     :R$
1820 IF R$="" THEN 1810 ELSE R=VAL(R$)
1830 IF R>0 AND R<=TR THEN 1860
```

```
1840 DISPLAY AT(23,3):"Impossible !"
1850 FOR DELAY=1 TO 2 :: CALL SOUND(1000,110,1)::
     NEXT DELAY :: GOTO 1800
1860 RETURN
1870 REM COLUMN
1880 DISPLAY AT(23,15):"Col ="
1890 ACCEPT AT(23,21)BEEP SIZE(2)VALIDATE(DIGIT,""
     ):C$
1900 IF C$="" THEN 1890 ELSE C=VAL(C$)
1910 IF C>0 AND C<=NC(R)THEN 1940
1920 DISPLAY AT(23,15):"Impossible !"
1930 FOR DELAY=1 TO 2 :: CALL SOUND(1000,110,1)::
     NEXT DELAY :: GOTO 1880
1940 RETURN
1950 REM TAKE BITE
1960 RW=R+6
1970 CL=2*C+3
1980 FOR I=RW TO TR+6
1990 DISPLAY AT(I,CL):""
2000 NEXT I
2010 REM BLANK-OUT ROW NUMBERS
2020 IF C<>1 THEN 2070
2030 FOR I=RW TO TR+6
2040 DISPLAY AT(I,2):""
2050 NEXT I
2060 REM BLANK-OUT COLUMN NUMBERS
2070 IF R=1 THEN DISPLAY AT(5,CL):""
2080 RETURN
2090 REM COUNTERS
2100 FOR I=R TO TR
2110 IF NC(I)>=C THEN NC(I)=C-1
2120 NEXT I
2130 IF C=1 THEN TR=R-1
2140 RETURN
2150 REM GAME OVER
2160 CALL CLEAR
2170 DISPLAY AT(1,1):"So sorry, ";NAME$(PLAYER);".
     "
2180 DISPLAY AT(3,1):"You ate the vanilla part of"
2190 DISPLAY AT(4,1):"the cookie."
2200 RETURN
```

Chapter 4

Curve-Fitting Routines

Curve-Fitting Routines

This chapter presents correlation and curve-fitting techniques designed to give you the kind of computational capability that, until recently, was available only on mainframe computers. Five programs are offered:

- "Simple Correlation Coefficients" estimates the linear association between variables.
- "Simple Least Squares" fits a trend line through a group of observations.
- "Multiple Linear-Regression Analysis" estimates the linear relationship between one variable and several other variables.
- "t-Curve Critical Values" tests hypotheses in regression analysis.
- "General-Form Curve Fitter" estimates any one of four different types of simple regression equations: linear, power, exponential, and reciprocal.

The first three programs require you to enter a set of observations on one or more variables. Don't worry if you enter an incorrect value. Each program includes a handy look-and-edit feature that lets you change any of your entries if need be.

In addition, in the programs for computing correlation coefficients and for performing multiple linear regression, you can change the preset combination of maximum allowable numbers of observations and variables in order to best use the available memory space. Instructions will be displayed on the screen.

All programs require 16K RAM. However, for large sets of data—for example, 100 observations on seven or eight variables—you'll probably need an additional 16K of RAM.

Correlation Coefficients

Most people probably view the track of gold and silver prices shown in Table 4-1 with a why-didn't-I-buy-then attitude. Regrets aside, a quick glance at the data suggests that the two sets of prices are strongly related. Both rose sharply in 1973 and 1974 and again in 1979 and 1980, and the time periods for lowest prices also coincide. But just how strong is this

seemingly robust relationship? One way to find out is to measure the linear correlation between the two sets of price data.

Table 4-1. Prices Per Troy Ounce of Gold and Silver in the United States

Year	Price of Gold	Price of Silver
1967	$ 35.0	$ 1.5
1968	39.0	2.1
1969	41.5	1.8
1970	36.2	1.8
1971	41.0	1.6
1972	58.1	1.7
1973	96.5	2.6
1974	158.1	4.8
1975	161.7	4.4
1976	125.9	4.4
1977	147.4	4.6
1978	192.9	5.4
1979	303.6	10.7
1980	618.0	21.6
1981	458.7	10.7
1982	378.5	8.0
June 83	407.9	11.6

Source: From prices and indices compiled by the Bureau of Labor

The degree of linear association between two variables is given by a linear correlation coefficient. Such coefficients are always between -1 and $+1$, inclusive. A value close to either extreme means the linear relationship between the two terms is strong. If the correlation is close to zero, however, then the relationship is weak. Figure 4-1 illustrates these relationships.

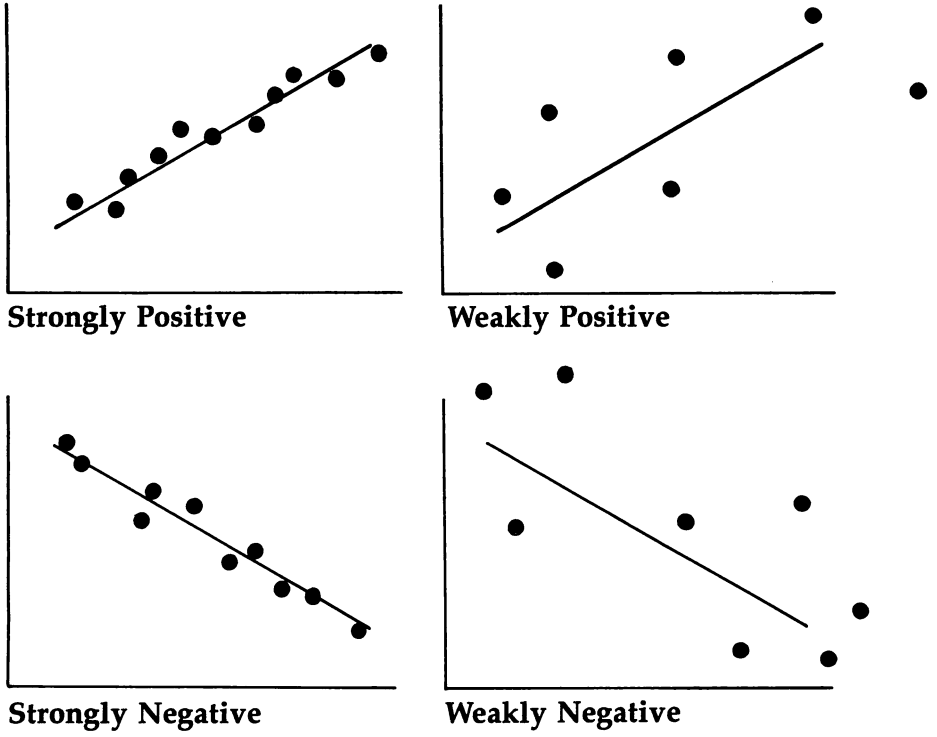
With the simple correlation coefficient program, it's easy to find such coefficients using your TI. Enter the data on gold and silver prices from 1967 through 1983, then review and edit the data before running the program. The final display should read:

SIMPLE CORRELATION COEFFICIENTS

	X1	X2
X1	1.000	0.960
X2	0.960	1.000

X1 represents the price of gold, and X2 represents the price of silver. The simple linear correlation between gold and silver prices is 0.960.

Figure 4-1. Types of Linear Correlation



A correlation of 0.960 is very high, implying a strong relationship between the variables. Such a strong relationship, either direct or inverse, may tempt you to call one thing the cause and the other the effect. At times this is reasonable, but at other times it can cause trouble. The fact that the sun rises

after the cock crows, for example, doesn't mean that the crowing causes the rising. Some third variable may cause both, or they may even be totally unrelated. Always rely on common sense or well-established theory to determine which correlations are reasonable and which are merely coincidence.

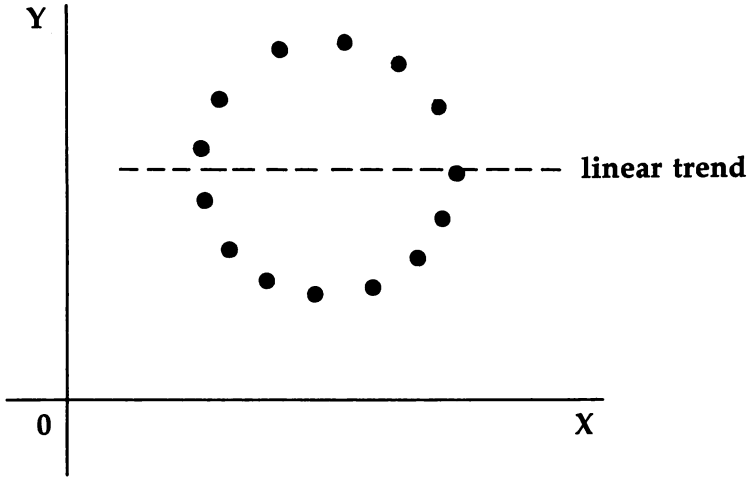
Note that if the observations on a variable are all the same, then any simple correlation coefficient involving that variable could not be computed since division by zero would be required. For instance, if X1 values of 3, 3, and 3 were compared to X2 values of 7, 8, and 20 (or to any values of X2, for that matter), the output would read:

CORRELATION COEFFICIENTS

	X1	X2
X1	UNDF'D	UNDF'D
X2	UNDF'D	1

An additional word of warning: Only *linear* association is measured by the correlation coefficient. There are other relationships that linear correlation measurements will not reveal. In a circle, for example, the linear relationship between Y and X is zero (as Figure 4-2 shows) while the circular association is perfect.

Figure 4-2. Circular Association



Finally, on a historical note, the idea of a correlation coefficient originated with the nineteenth century Englishman Sir Francis Galton, cousin of Charles Darwin. Historian James Newman tells us that Galton got the idea for an index of correlation one morning while waiting for a train. It took a few years, but Galton's Index of Correlation has evolved into what the TI now computes as the simple correlation coefficient.

Program 4-1. Simple Correlation Coefficients

```
100 REM CORRELATION COEFFICIENTS
110 REM
120 REM
130 REM INITIALIZE
140 GOSUB 240
150 REM ENTER DATA
160 GOSUB 630
170 REM EDIT DATA
180 GOSUB 930
190 REM COMPUTE COEFFICIENTS
200 GOSUB 1270
210 REM DISPLAY RESULTS
220 GOSUB 1490
230 END
240 REM INITIALIZE
250 REM MAXIMUM NUMBERS OF OBSERVATIONS & VARIABLE
    S
260 GOSUB 320
270 REM ENTER NUMBER OF VARIABLES
280 GOSUB 490
290 REM CREATE VARIABLE SYMBOLS
300 GOSUB 580
310 RETURN
320 REM MAXIMUM VALUES
330 DATA 25,15
340 DIM V$(15),R(15,30),S(15),SS(15),X(25,15)
350 READ MAXN,MAXV
360 CALL CLEAR
370 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES SIMPLE"
380 DISPLAY AT(2,1):"CORRELATION COEFFICIENTS FOR
    THE"
390 DISPLAY AT(3,1):"VARIABLES YOU ENTER."
400 DISPLAY AT(5,1):"THE MAXIMUM NUMBERS OF"
410 DISPLAY AT(6,1):"VARIABLES AND OBSERVATIONS"
420 DISPLAY AT(7,1):"ALLOWED ARE:"
430 DISPLAY AT(9,3):"VARIABLES{4 SPACES}= ";MAXV
440 DISPLAY AT(10,3):"OBSERVATIONS = ";MAXN
```

```
450 DISPLAY AT(15,1):"CHANGE LINES 330 & 340"
460 DISPLAY AT(16,1):"FOR DIFFERENT LIMITS."
470 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE" ::
    ACCEPT AT(23,25):Z$
480 RETURN
490 REM ENTER # OF VARIABLES
500 CALL CLEAR
510 DISPLAY AT(1,1):"HOW MANY VARIABLES DO"
520 DISPLAY AT(2,1):"YOU HAVE ?"
530 ACCEPT AT(2,11)VALIDATE(DIGIT,"")BEEP:K$
540 IF K$="" THEN 530 ELSE K=VAL(K$)
550 IF K<2 THEN DISPLAY AT(24,1):"AT LEAST 2 VARIA
    BLES NEEDED" :: GOTO 530
560 IF K>MAXV THEN DISPLAY AT(24,1):"ONLY";MAXV;"A
    LLOWED." :: GOTO 530
570 RETURN
580 REM CREATE SYMBOLS
590 FOR I=1 TO K
600 V$(I)="X"&STR$(I)
610 NEXT I
620 RETURN
630 REM ENTER DATA
640 REM ON THE FIRST VARIABLE
650 GOSUB 690
660 REM ON THE OTHERS
670 GOSUB 830
680 RETURN
690 REM ENTER DATA ON FIRST VARIABLE
700 CALL CLEAR
710 DISPLAY AT(1,1):"PLEASE ENTER OBSERVATIONS"
720 DISPLAY AT(2,1):"FOR ";V$(1);". HIT 'ENTER' WH
    EN"
730 DISPLAY AT(3,1):"THROUGH."
740 N=MAXN
750 FOR J=1 TO MAXN
760 DISPLAY AT(5,1):V$(1);"(";J;")= "
770 ACCEPT AT(5,11)VALIDATE(NUMERIC,""):X$
780 IF X$="" THEN N=J-1 :: J=MAXN ELSE X(J,1)=VAL(
    X$)
790 NEXT J
800 REM CHECK FOR TOO FEW OBSERVATIONS
810 IF N<2 THEN DISPLAY AT(23,1):"AT LEAST TWO OBS
    ERVATIONS" :: DISPLAY AT(24,1):"ARE REQUIRED.
    TRY AGAIN." :: GOTO 740
820 RETURN
830 REM ENTER DATA FOR OTHER VARIABLES
840 FOR I=2 TO K
850 CALL CLEAR
860 DISPLAY AT(1,1):"PLEASE ENTER OBSERVATIONS"
```

```
870 DISPLAY AT(2,1):"FOR ";V$(I);"."
880 FOR J=1 TO N
890 DISPLAY AT(5,1):V$(I);"(";J;")= "
900 ACCEPT AT(5,11)VALIDATE(NUMERIC):X(J,I)
910 NEXT J :: NEXT I
920 RETURN
930 REM EDIT DATA
940 FOR I=1 TO K
950 FOR L=0 TO INT((N-1)/10)
960 REM DISPLAY DATA
970 GOSUB 1030
980 REM CORRECT DATA
990 GOSUB 1110
1000 NEXT L
1010 NEXT I
1020 RETURN
1030 REM DISPLAY DATA
1040 CALL CLEAR
1050 DISPLAY AT(1,1):"THESE ARE VALUES OF ";V$(I);
     ":"
1060 FOR J=1 TO 10
1070 M=J+L*10
1080 IF M<=N THEN DISPLAY AT(J+2,1):V$(I);TAB(4);"
     (";M;TAB(9);")= ";X(M,I)
1090 NEXT J
1100 RETURN
1110 REM CORRECT DATA
1120 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1130 ACCEPT AT(16,20)VALIDATE("YN","")BEEP:S$
1140 IF S$="N" THEN 1260
1150 IF S$="" THEN 1130
1160 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1170 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
1180 ACCEPT AT(19,24)BEEP VALIDATE(NUMERIC,""):Q$
1190 IF Q$="" THEN 1180 ELSE Q=INT(VAL(Q$))
1200 IF Q<(1+L*10)OR Q>N OR Q>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER NUMBER" :: DISPLAY A
     T(22,1):"WITHIN BOUNDS SHOWN." :: GOTO 1180
1210 DISPLAY AT(21,1):"WHAT SHOULD THE"
1220 DISPLAY AT(22,1):"VALUE BE ?"
1230 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1240 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
1250 GOSUB 1030 :: GOTO 1120
1260 RETURN
1270 REM COMPUTE COEFFICIENTS
1280 CALL CLEAR
1290 DISPLAY AT(1,1):"COMPUTING ..."
1300 REM SUM OF OBSERVATIONS(S)
1310 REM SUM OF SQUARED OBSERVATIONS(SS)
```

```
1320 FOR I=1 TO K
1330 S(I)=0 :: SS(I)=0
1340 FOR J=1 TO N
1350 S(I)=S(I)+X(J,I):: SS(I)=SS(I)+X(J,I)*X(J,I)
1360 NEXT J :: NEXT I
1370 REM COMPUTE SIMPLE CORRELATION MATRIX
1380 FOR I=1 TO K
1390 FOR J=I TO K
1400 SC=0
1410 FOR L=1 TO N :: SC=SC+X(L,I)*X(L,J):: NEXT L
1420 SQ=SQR((N*SS(I)-S(I)*S(I))*(N*SS(J)-S(J)*S(J)
     ))
1430 IF SQ<>0 THEN R(I,J)=(N*SC-S(I)*S(J))/SQ ELSE
      R(I,J)=-999
1440 R(J,I)=R(I,J)
1450 IF SQ<>0 THEN R(I,I)=1
1460 NEXT J
1470 NEXT I
1480 RETURN
1490 REM PRINT COEFFICIENTS IN 10 BY 3 BLOCKS
1500 FOR Q=1 TO K STEP 10
1510 GOSUB 1540
1520 NEXT Q
1530 RETURN
1540 REM PRINT
1550 IMAGE ###.###
1560 FOR I=1 TO K STEP 3
1570 CALL CLEAR :: CALL HCHAR(1,1,61,28)
1580 DISPLAY AT(2,10):"SIMPLE"
1590 DISPLAY AT(3,1):"CORRELATION COEFFICIENTS" ::
      CALL HCHAR(4,1,61,28)
1600 REM PRINT COLUMN HEADING
1610 ROW=8 :: COL=8
1620 FOR L=I TO I+2
1630 IF L<=K THEN DISPLAY AT(6,COL):V$(L)
1640 COL=COL+7
1650 NEXT L
1660 REM PRINT COEFFICIENTS
1670 FOR J=Q TO Q+9
1680 IF J>K THEN 1770
1690 DISPLAY AT(ROW,1):V$(J)
1700 COL=4
1710 FOR L=I TO I+2
1720 IF L<=K AND R(J,L)>-999 THEN DISPLAY AT(ROW,C
     OL):USING 1550:R(J,L)
1730 IF L<=K AND R(J,L)=-999 THEN DISPLAY AT(ROW,C
     OL):" UNDF'D"
1740 COL=COL+7
1750 NEXT L
```

```
1760 ROW=ROW+1
1770 NEXT J
1780 CALL HCHAR(23,1,61,28)
1790 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1800 ACCEPT AT(24,25):Z$
1810 NEXT I
1820 RETURN
```

Simple Least Squares

Simple linear-regression analysis is a statistical technique for trying to explain changes in one variable in terms of another variable. This technique is often used to examine the effects of public policy, to forecast, and to test hypotheses on the causes of economic, political, and other social and physical phenomena.

Suppose you want to estimate the relationship between stock prices and interest rates. Your guess is that stock prices rise when interest rates fall and fall when interest rates rise.

Table 4-2. Stock Prices and Interest Rates

Year and Month	Standard and Poor's Index of 500 Leading Stocks	Three-Month T-Bill Rate
82:1	100.0	12.3
:2	97.6	13.5
:3	94.5	12.7
:4	99.2	12.7
:5	99.2	12.1
:6	93.5	12.5
:7	93.3	11.4
:8	93.5	8.7
:9	104.4	7.9
:10	113.1	7.7
:11	117.8	8.1
:12	118.8	7.9
83:1	123.0	7.9
:2	125.2	8.1
:3	129.5	8.4
:4	134.5	8.2
:5	139.9	8.2
:6	141.9	8.8

To check that guess, use the Simple Least-Squares program. Using the sample data given in Table 4-2, you'll get the following display:

REGRESSION RESULTS

TERM	ESTIMATED VALUE	STANDARD ERROR
B0	165.94547	13.833
B1	-5.46651	1.374
NUMBER OF OBSERVATIONS	=	18.000
R-SQUARED	=	0.497
STANDARD ERROR OF THE ESTIMATE	=	12.455
F STATISTIC	=	15.830

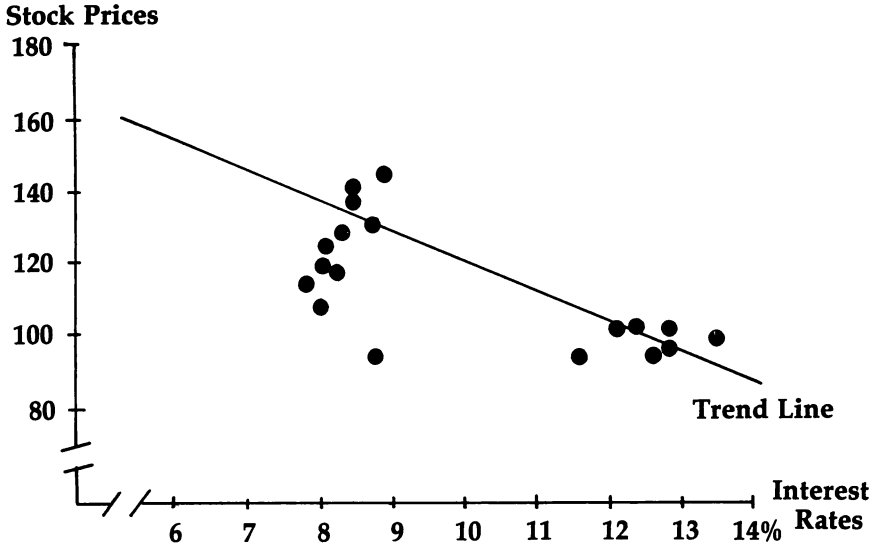
Regression results are not difficult to interpret. The values of B0 and B1 describe a straight line depicting the relationship between the two variables. B0 is the constant term and B1 the slope. This estimated regression equation, or trend line, is depicted in Figure 4-3. Since B1, the slope, has a value of −5.46651, you could expect stock prices to fall by almost 5.5 points for every one-point increase in interest rates. The inverse relationship between stock prices and interest rates holds, just as you would suspect.

Next, the standard error of the estimate, about 12.5 units in this case, is the standard deviation of actual minus predicted Y values. It can be loosely interpreted as the average error made in predicting Y using the regression equation. Finally, the F-statistic measures how well the regression equation explains the variation in Y.

Some applications of regression analysis become much more complicated than the one given here. Nevertheless, many regression problems do not require a fancy solution and can be handled with a simple least-squares routine.

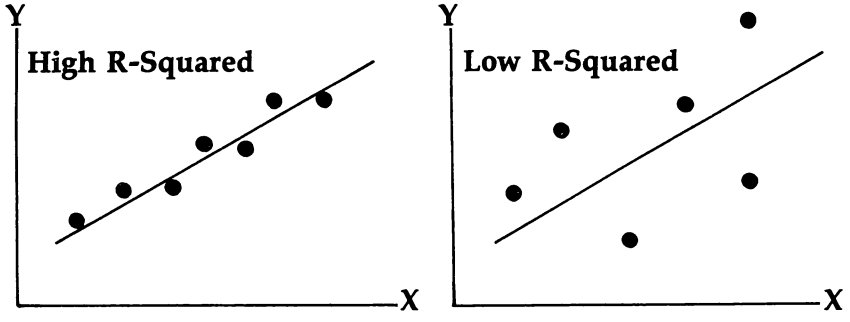
The standard errors of B0 and B1 indicate the precision of the estimated values of these two parameters. The smaller the standard error, the more precise is the estimate.

Figure 4-3. Regression Equation



The last three figures are goodness-of-fit statistics. R-squared, or the coefficient of determination, is the proportion of variation in Y (stock prices) explained by X (interest rates). It ranges from 0 to 1, with a value close to 0 indicating a weak association and a value close to 1 indicating a strong association. Figure 4-4 illustrates these relationships. In this example, roughly half of the change in stock prices is due to interest rates.

Figure 4-4. Goodness of Fit



Notes on the Program

In a simple linear-regression analysis the term to be explained or predicted is called the dependent variable. The term that does the explaining is called, appropriately enough, the explanatory variable. *Simple* means that only one explanatory variable appears in the regression equation (two or more appear in multiple regression), and *linear* refers to the straight-line nature of the relationship. Hence, simple linear regression involves estimating a straight line through a set of observations on a dependent and an explanatory variable.

The equation for a straight line is $Y = a + bX$. Y usually is plotted on the vertical axis, while X is plotted on the horizontal axis. The symbol *a* represents the Y-intercept of the line, and *b* represents its slope. The slope *b* can be defined as the change in Y divided by the corresponding change in X.

The equation for the simple linear-regression model is similar to that for a straight line. It takes the following form:

Y = a + bX + random error

Y is the dependent variable; X is the explanatory variable; and a and b are, as before, the Y-intercept and slope. The only difference between the two equations is the random error term.

If the error term were absent from an equation, regression analysis would not be needed since each pair of X and Y observations would lie exactly on a straight line. For example, if you let X equal the total number of hits and Y equal the total times at bat, the relationship between hits and times at bat—the batting average—graphs as a straight line. But for many other relationships, the association between X and Y is less exact. As a result, the random error term must be present in the equation.

In essence, the simple linear-regression routine identifies the line which minimizes the sum of squared distances of observations from that line. This is equivalent to drawing, with computer precision, a line through a plot of points to best reflect the apparent trend.

Program 4-2. Simple Least Squares

```
100 REM SIMPLE LINEAR REGRESSION
130 REM INITIALIZE
140 GOSUB 240
150 REM ENTER DATA
```

```
160 GOSUB 410
170 REM EDIT DATA
180 GOSUB 720
190 REM TALLY STATISTICS
200 GOSUB 1060
210 REM PRINT STATISTICS
220 GOSUB 1410
230 END
240 REM INITIALIZE
250 CALL CLEAR
260 DISPLAY AT(1,1):"  SIMPLE LINEAR REGRESSION"
270 REM MAXIMUM NUMBER OF OBSERVATIONS
280 DATA 99
290 READ NMAX
300 DIM X(99,2)
310 V$(1)="Y" :: V$(2)="X"
320 DISPLAY AT(4,1):"A TOTAL OF";NMAX;"OBSERVATION
    S"
330 DISPLAY AT(5,1):"ON YOUR REGRESSION EQUATION"
340 DISPLAY AT(6,1):"ARE ALLOWED."
350 DISPLAY AT(8,1):"FOR A DIFFERENT LIMIT,"
360 DISPLAY AT(9,1):"PLEASE CHANGE LINES 280"
370 DISPLAY AT(10,1):"AND 300."
380 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
390 ACCEPT AT(24,24):Z$
400 RETURN
410 REM ENTER DATA
420 REM ON Y
430 GOSUB 470
440 REM ON X
450 GOSUB 620
460 RETURN
470 REM ENTER DATA ON Y
480 CALL CLEAR
490 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
500 DISPLAY AT(2,1):"DEPENDENT VARIABLE (Y)."
510 DISPLAY AT(3,1):"HIT 'ENTER' WHEN THROUGH."
520 N=NMAX
530 FOR I=1 TO NMAX
540 DISPLAY AT(8,1):"Y(";I;TAB(8);")= ?"
550 ACCEPT AT(8,13)VALIDATE(NUMERIC,""):S$
560 IF S$="" THEN N=I-1 :: I=NMAX :: GOTO 580
570 X(I,1)=VAL(S$)
580 NEXT I
590 REM CHECK FOR TOO FEW OBSERVATIONS
600 IF N<3 THEN DISPLAY AT(23,1):"AT LEAST 3 OBSER
    VATIONS ARE" :: DISPLAY AT(24,1):"NEEDED. PLEA
    SE TRY AGAIN." :: GOTO 520
610 RETURN
```

```
620 REM ENTER DATA ON X
630 CALL CLEAR
640 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
650 DISPLAY AT(2,1):"EXPLANATORY VARIABLE (X)."
660 FOR I=1 TO N
670 DISPLAY AT(7,1):"X(";I;TAB(8);")= ?"
680 ACCEPT AT(7,13)VALIDATE(NUMERIC,""):S$
690 IF S$="" THEN X(I,2)=0 ELSE X(I,2)=VAL(S$)
700 NEXT I
710 RETURN
720 REM EDIT DATA
730 FOR I=1 TO 2
740 FOR L=0 TO INT((N-1)/10)
750 REM DISPLAY DATA
760 GOSUB 820
770 REM CORRECT DATA
780 GOSUB 900
790 NEXT L
800 NEXT I
810 RETURN
820 REM DISPLAY DATA
830 CALL CLEAR
840 DISPLAY AT(1,1):"THESE ARE VALUES OF ";V$(I);"
    :"
850 FOR J=1 TO 10
860 Q=J+L*10
870 IF Q<=N THEN DISPLAY AT(J+2,1):V$(I);"(";Q;TAB
    (8);")= ";X(Q,I)
880 NEXT J
890 RETURN
900 REM CORRECT DATA
910 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
920 ACCEPT AT(16,20)VALIDATE("YN","")BEEP:S$
930 IF S$="N" THEN 1050
940 IF S$="" THEN 920
950 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
960 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
970 ACCEPT AT(19,24)BEEP VALIDATE(NUMERIC,""):Q$
980 IF Q$="" THEN 970 ELSE Q=INT(VAL(Q$))
990 IF Q<(1+L*10)OR Q>N OR Q>(10+L*10)THEN DISPLAY
     AT(21,1):"PLEASE ENTER NUMBER WITHIN" :: DISP
    LAY AT(22,1):"BOUNDS SHOWN." :: GOTO 970
1000 DISPLAY AT(21,1):"WHAT SHOULD THE"
1010 DISPLAY AT(22,1):"VALUE BE ?"
1020 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1030 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
1040 GOSUB 830 :: GOTO 910
1050 RETURN
1060 REM TALLY STATISTICS
```

```
1070 CALL CLEAR
1080 DISPLAY AT(1,1):"TALLYING STATISTICS ..."
1090 REM KEY SUMS
1100 GOSUB 1160
1110 REM COEFFICIENT & CONSTANT
1120 GOSUB 1260
1130 REM ANOVA TERMS
1140 GOSUB 1300
1150 RETURN
1160 REM KEY SUMS
1170 SX=0 :: SY=0 :: XQ=0 :: YQ=0 :: CP=0
1180 FOR I=1 TO N
1190 SX=SX+X(I,2)
1200 SY=SY+X(I,1)
1210 XQ=XQ+X(I,2)*X(I,2)
1220 YQ=YQ+X(I,1)*X(I,1)
1230 CP=CP+X(I,1)*X(I,2)
1240 NEXT I
1250 RETURN
1260 REM COEFFICIENT & CONSTANT
1270 B=(N*CP-SX*SY)/(N*XQ-SX*SX)
1280 A=(SY-B*SX)/N
1290 RETURN
1300 REM ANOVA TERMS
1310 TSS=YQ-SY*SY/N
1320 RSS=B*(CP-SX*SY/N)
1330 ESS=TSS-RSS
1340 REM ERROR VARIANCE & STANDARD ERROR OF THE ES
     TIMATE
1350 EV=ESS/(N-2)
1360 SEE=SQR(EV)
1370 REM STANDARD ERRORS OF THE COEFFICIENT & CONS
     TANT
1380 SB=SQR(EV/(XQ-SX*SX/N))
1390 SA=SQR(EV*XQ/(N*XQ-SX*SX))
1400 RETURN
1410 REM PRINT STATISTICS
1420 REM EQUATION
1430 GOSUB 1470
1440 REM ANOVA TERMS
1450 GOSUB 1630
1460 RETURN
1470 REM EQUATION
1480 CALL CLEAR
1490 CALL HCHAR(1,3,61,28)
1500 DISPLAY AT(2,6):"REGRESSION RESULTS"
1510 CALL HCHAR(3,3,61,28)
1520 DISPLAY AT(5,9):"ESTIMATED";TAB(21);"STANDARD
     "
```

```
1530 DISPLAY AT(6,1):"TERM";TAB(13);"VALUE";TAB(24
     );"ERROR"
1540 IMAGE #########.#####
1550 IMAGE #######.###
1560 DISPLAY AT(8,1):"BØ"
1570 DISPLAY AT(8,3):USING 1540:A
1580 DISPLAY AT(8,18):USING 1550:SA
1590 DISPLAY AT(9,1):"B1"
1600 DISPLAY AT(9,3):USING 1540:B
1610 DISPLAY AT(9,18):USING 1550:SB
1620 RETURN
1630 REM ANOVA TERMS
1640 IMAGE =########.###
1650 DISPLAY AT(12,1):"NUMBER OF" :: DISPLAY AT(13
     ,1):"OBSERVATIONS" :: DISPLAY AT(13,16):USING
      1640:N
1660 DISPLAY AT(15,1):"R-SQUARED" :: DISPLAY AT(15
     ,16):USING 1640:RSS/TSS
1670 DISPLAY AT(17,1):"STANDARD ERROR" :: DISPLAY
     AT(18,1):"OF THE ESTIMATE"
1680 DISPLAY AT(18,16):USING 1640:SEE
1690 DISPLAY AT(20,1):"F-STATISTIC" :: DISPLAY AT(
     20,16):USING 1640:RSS/EV
1700 CALL HCHAR(23,3,61,28)
1710 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1720 ACCEPT AT(24,25):Z$
1730 RETURN
```

Multiple Linear-Regression Analysis

The number of traffic fatalities in the United States varies from locality to locality. In Virginia, for example, 1050 people died on the highways in 1964. During this same year, however, 118 died in Delaware.

Why the difference? To try to find out, use the Multiple Linear-Regression Analysis program. First, list any variables that might explain a state's fatality count—the number of automobile drivers in a state, for instance, or a state's rural road mileage. You could reasonably expect fatalities to be highest in those states with many drivers and a lot of rural roads. Next, gather observations on the chosen variables. Table 4-3 summarizes those observations for 14 states and the District of Columbia. Finally, enter the data into the computer.

The program first asks for the number of explanatory variables, or X's. Enter 2, since you are using two terms to explain

Table 4-3. Automobile Fatalities in the United States

State	Deaths in 1964	Drivers (10,000)	Rural Road Mileage (1000)
Delaware	118	30	3.4
Washington, D.C.	115	35	0
Indiana	1410	254	89
Iowa	833	150	100
Kansas	669	136	124
Maryland	616	157	29
Massachusetts	766	255	17
Mississippi	648	85	59
Nevada	215	23	44
New Mexico	387	54	62
South Dakota	270	40	87
Utah	295	57	32
Vermont	131	20	13
Virginia	1050	208	50
Washington	730	160	59

Source: "Car accidents—environmental aspects," by D. F. Andrews, *International Statistical Review*, Vol. 41, 1973, as reprinted in Draper and Smith, *Applied Regression Analysis*, 2nd edition, 1981, p. 191.

traffic fatalities: the total number of drivers and the total rural road mileage.

The TI then asks you to enter values for the dependent variable Y. Those values, actually numbers of traffic fatalities, also come from Table 4-3. Type in 118 for Delaware, 115 for Washington, D.C., and so on, right down the list.

After a few seconds the display will read:

REGRESSION EQUATION

TERM	ESTIMATED VALUE	STANDARD ERROR
B0	-11.329	66.715
B1	3.778	0.432
B2	2.780	0.976

The symbol B0 stands for the constant term in our regression equation. The symbols B1 and B2 correspond to our two X's, number of drivers (X1) and rural road mileage (X2). Both B1 and B2 measure the impact on Y of a one-unit (that is, 10,000 drivers) change in the value of the corresponding X. The B1 value suggests that 3.778 more people are expected to die in auto accidents over the course of a year as the number of drivers in a state increases by 10,000. This is illustrated in Figure 4-5. Similarly, the value of B2 suggests that 2.780 additional fatalities per year can be expected for each additional 1000 miles of rural roads.

Instruct the TI to continue. It displays:

SUMMARY STATISTICS

	SUMS OF SQUARES	DEGREES OF FREEDOM
TOTAL	2010494.400	14
REG'N	1812250.132	2
ERROR	198244.268	12
R-SQUARED	=	0.901
R BAR SQUARED	=	0.885
F STATISTIC	=	54.849
ERROR VARIANCE	=	16520.356
STANDARD ERROR OF THE ESTIMATE	=	128.532

Some of these statistics are familiar from the simple linear-regression example, but total, regression (Reg'n), and error sum of squares may be new to you. The total sum of squares measures the dispersion of observations on Y about the mean, while the regression and error sums of squares measure, respectively, the dispersion explained and unexplained by the X's.

The R-squared value, 0.901, indicates that 90.1 percent of the variation in traffic fatalities from state to state is explained by the equation. R Bar Squared (which may be negative) is the R-squared statistic adjusted for degrees of freedom. It is useful in assessing the relative strengths of different models for explaining the same Y.

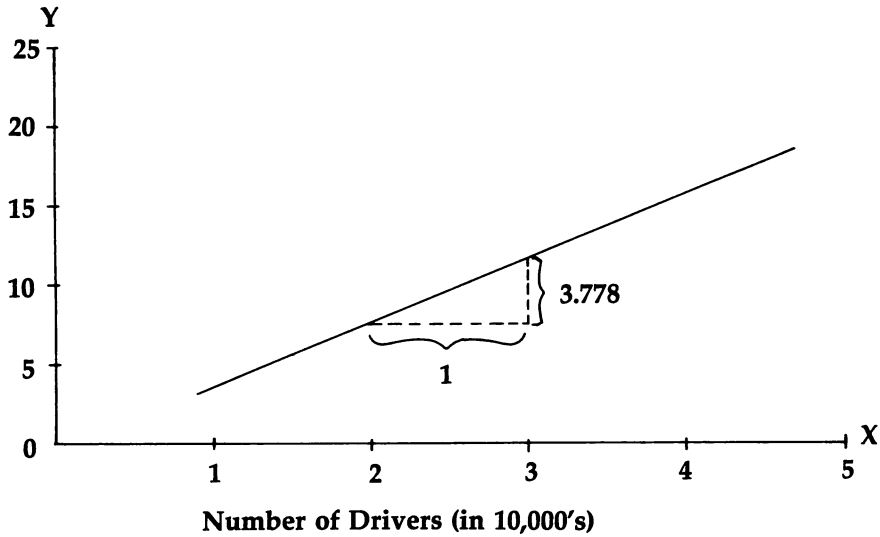
Two final statistics are also displayed:

CORRELATION COEFFICIENT = −0.110
DURBIN-WATSON STATISTIC = 2.214

The first measures the degree of linear relationship between successive regression residuals; the second is used to test for this kind of correlation.

Multiple linear-regression analysis is a straightforward extension of the simple linear case. Simple linear regression considers only one explanatory variable and fits a straight line between the values of X and Y, while multiple regression considers two or more explanatory variables to estimate a plane instead of a line. The difference between simple and multiple regression is much like the difference between singles and doubles tennis. There are more players (X's) in doubles, but the basic game—statistics—remains the same.

Figure 4-5. Traffic Fatalities and Number of Drivers



A one-unit increase in the number of drivers in a state means that 3.778 more people will die on the highways over the course of a year, if everything else stays the same. The slope of the line, or the change in Y divided by the change in X, is therefore 3.778.

Notes on the Program

1. The maximum allowable numbers of observations and explanatory variables (X's) are 50 and 4, respectively, or N and K in general. As the display indicates, change lines 330 and 340 for different values. Line 330 is a DATA statement containing N and K values. Line 340 is a DIMension statement. In changing line 340, be sure to use the value K + 1 instead of K, where appropriate. The extra 1 is for the constant term.

2. Regression coefficients are tallied using a highly accurate technique in modern numerical analysis called Modified Gram-Schmidt Orthogonalization. The TI-99 version of this algorithm yields more accurate answers than do many popular statistical packages executed on mainframe computers.

Program 4-3. Multiple Linear-Regression Analysis

```
100 REM MULTIPLE LINEAR REGRESSION
130 REM INITIALIZE
140 GOSUB 240
150 REM ENTER DATA
160 GOSUB 670
170 REM EDIT DATA
180 GOSUB 980
190 REM TALLY STATISTICS
200 GOSUB 1310
210 REM PRINT RESULTS
220 GOSUB 2590
230 END
240 REM INITIALIZE
250 REM MAXIMUM NUMBERS OF OBSERVATIONS & VARIABLE
    S
260 GOSUB 320
270 REM ENTER NUMBER OF VARIABLES
280 GOSUB 500
290 REM CREATE VARIABLE SYMBOLS
300 GOSUB 600
310 RETURN
320 REM MAXIMUM VALUES
330 DATA 50,4
340 DIM B(5),C(5),B$(5),V$(5),R(5,5),V(5,5),Q(50,5
    ),X(50,5)
350 READ MAXN,MAXV
360 CALL CLEAR
370 DISPLAY AT(1,1):"THIS PROGRAM ESTIMATES A"
380 DISPLAY AT(2,1):"MULTIPLE LINEAR REGRESSION"
```

```
390 DISPLAY AT(3,1):"EQUATION."
400 DISPLAY AT(5,1):"THE MAXIMUM NUMBERS OF"
410 DISPLAY AT(6,1):"EXPLANATORY VARIABLES (X'S)"
420 DISPLAY AT(7,1):"& OBSERVATIONS ALLOWED ARE:"
430 DISPLAY AT(10,12):"X'S = ";MAXV
440 DISPLAY AT(11,3):"OBSERVATIONS = ";MAXN
450 DISPLAY AT(15,1):"CHANGE LINES 330 & 340"
460 DISPLAY AT(16,1):"FOR DIFFERENT LIMITS."
470 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
480 ACCEPT AT(23,25):Z$
490 RETURN
500 REM NUMBER OF X'S
510 CALL CLEAR
520 DISPLAY AT(1,1):"HOW MANY EXPLANATORY"
530 DISPLAY AT(2,1):"VARIABLES (X'S) DO"
540 DISPLAY AT(3,1):"YOU HAVE ? "
550 ACCEPT AT(3,12)VALIDATE(DIGIT,"")BEEP:K$
560 IF K$="" THEN 550 ELSE K=VAL(K$)
570 IF K<1 THEN DISPLAY AT(24,1):"AT LEAST ONE X I
    S NEEDED" :: GOTO 550
580 IF K>MAXV THEN DISPLAY AT(24,1):"ONLY";MAXV;"A
    LLOWED" :: GOTO 550
590 RETURN
600 REM SYMBOLS
610 V$(0)="Y" :: B$(0)="B0"
620 FOR I=1 TO K
630 V$(I)="X"&STR$(I)
640 B$(I)="B"&STR$(I)
650 NEXT I
660 RETURN
670 REM ENTER DATA
680 REM ON Y
690 GOSUB 730
700 REM ON THE X'S
710 GOSUB 880
720 RETURN
730 REM Y
740 CALL CLEAR
750 DISPLAY AT(1,1):"PLEASE ENTER OBSERVATIONS"
760 DISPLAY AT(2,1):"ON THE DEPENDENT VARIABLE."
770 DISPLAY AT(3,1):"HIT 'ENTER' WHEN THROUGH."
780 N=MAXN
790 FOR J=1 TO MAXN
800 DISPLAY AT(5,1):V$(0);"(";J;")= "
810 ACCEPT AT(5,11)VALIDATE(NUMERIC,""):S$
820 IF S$="" THEN N=J-1 :: J=MAXN ELSE X(J,0)=VAL(
    S$)
830 NEXT J
840 REM DEGREES OF FREEDOM
```

```
850 V1=N-K-1
860 IF V1<1 THEN DISPLAY AT(23,1):"YOU HAVE ";V1;"
    DEGREES OF" :: DISPLAY AT(24,1):"FREEDOM. PLEA
    SE TRY AGAIN." :: GOTO 780
870 RETURN
880 REM X'S
890 FOR I=1 TO K
900 CALL CLEAR
910 DISPLAY AT(1,1):"PLEASE ENTER OBSERVATIONS"
920 DISPLAY AT(2,1):"FOR ";V$(I);"."
930 FOR J=1 TO N
940 DISPLAY AT(5,1):V$(I);"(";J;")= "
950 ACCEPT AT(5,11)VALIDATE(NUMERIC):X(J,I)
960 NEXT J :: NEXT I
970 RETURN
980 REM EDIT DATA
990 FOR I=0 TO K
1000 FOR L=0 TO INT((N-1)/10)
1010 REM DISPLAY DATA
1020 GOSUB 1070
1030 REM CORRECT DATA
1040 GOSUB 1150
1050 NEXT L :: NEXT I
1060 RETURN
1070 REM DISPLAY DATA
1080 CALL CLEAR
1090 DISPLAY AT(1,1):"THESE ARE VALUES OF ";V$(I);
     ":"
1100 FOR J=1 TO 10
1110 Q1=J+L*10
1120 IF Q1<=N THEN DISPLAY AT(J+2,1):V$(I);"(";Q1;
     TAB(8);")= ";X(Q1,I)
1130 NEXT J
1140 RETURN
1150 REM CORRECT DATA
1160 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1170 ACCEPT AT(16,20)VALIDATE("YN","")BEEP:S$
1180 IF S$="N" THEN 1300
1190 IF S$="" THEN 1170
1200 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1210 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
1220 ACCEPT AT(19,24)BEEP VALIDATE(DIGIT,""):M$
1230 IF M$="" THEN 1220 ELSE M=VAL(M$)
1240 IF M<(1+L*10)OR M>N OR M>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER NUMBER WITHIN" :: DI
     SPLAY AT(22,1):"BOUNDS SHOWN." :: GOTO 1220
1250 DISPLAY AT(21,1):"WHAT SHOULD THE"
1260 DISPLAY AT(22,1):"VALUE BE ?"
1270 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
```

```
1280 IF S$="" THEN X(M,I)=0 ELSE X(M,I)=VAL(S$)
1290 GOSUB 1080 :: GOTO 1160
1300 RETURN
1310 REM TALLY
1320 REM PERFORM ORTHOGONALIZATION
1330 GOSUB 1410
1340 REM BACK-SOLVE FOR COEFFICIENTS
1350 GOSUB 2020
1360 REM COMPUTE VAR-COV MATRIX
1370 GOSUB 2140
1380 REM TALLY SUMMARY TERMS
1390 GOSUB 2820
1400 RETURN
1410 REM ORTHOGONALIZATION
1420 CALL CLEAR
1430 DISPLAY AT(1,1):"TALLYING STATISTICS ..."
1440 REM INSERT CONSTANT TERM
1450 GOSUB 1600
1460 K=K+1
1470 FOR Z=1 TO K
1480 REM COMPUTE KEY ELEMENT OF R
1490 GOSUB 1690
1500 REM COMPUTE COLUMN OF Q
1510 GOSUB 1760
1520 REM COMPUTE COLUMN OF R
1530 GOSUB 1810
1540 REM COMPUTE ELEMENT OF C
1550 GOSUB 1890
1560 REM REVISE X
1570 GOSUB 1950
1580 NEXT Z
1590 RETURN
1600 REM INSERT CONSTANT
1610 FOR I=K TO 1 STEP -1
1620 FOR J=1 TO N
1630 X(J,I+1)=X(J,I)
1640 NEXT J :: NEXT I
1650 FOR J=1 TO N
1660 X(J,1)=1
1670 NEXT J
1680 RETURN
1690 REM KEY ELEMENT OF R
1700 S=0
1710 FOR I=1 TO N
1720 S=S+X(I,Z)*X(I,Z)
1730 NEXT I
1740 R(Z,Z)=SQR(S)
1750 RETURN
1760 REM COLUMN OF Q
```

```
1770 FOR I=1 TO N
1780 Q(I,Z)=X(I,Z)/R(Z,Z)
1790 NEXT I
1800 RETURN
1810 REM COLUMN OF R
1820 IF Z=K THEN 1880
1830 FOR L=Z+1 TO K
1840 R(Z,L)=0
1850 FOR I=1 TO N
1860 R(Z,L)=R(Z,L)+X(I,L)*Q(I,Z)
1870 NEXT I :: NEXT L
1880 RETURN
1890 REM ELEMENT OF C
1900 C(Z)=0
1910 FOR I=1 TO N
1920 C(Z)=C(Z)+X(I,0)*Q(I,Z)
1930 NEXT I
1940 RETURN
1950 REM REVISE X
1960 IF Z=K THEN 2010
1970 FOR I=1 TO N
1980 FOR L=Z+1 TO K
1990 X(I,L)=X(I,L)-Q(I,Z)*R(Z,L)
2000 NEXT L :: NEXT I
2010 RETURN
2020 REM BACK-SOLVE
2030 B(K)=C(K)/R(K,K)
2040 FOR I=K-1 TO 1 STEP -1
2050 REM LEFT-SIDE SUM
2060 S=0
2070 FOR J=I+1 TO K
2080 S=S+R(I,J)*B(J)
2090 NEXT J
2100 REM COEFFICIENT
2110 B(I)=(C(I)-S)/R(I,I)
2120 NEXT I
2130 RETURN
2140 REM VAR-COV MATRIX
2150 REM COMPUTE ERROR VARIANCE
2160 GOSUB 2220
2170 REM INVERT MATRIX R
2180 GOSUB 2380
2190 REM COMPUTE UNSCALED VAR-COV MATRIX
2200 GOSUB 2510
2210 RETURN
2220 REM ERROR VARIANCE
2230 REM RESIDUALS = Y - Q*C
2240 FOR I=1 TO N
2250 S=0
```

```
2260 FOR J=1 TO K
2270 S=S+Q(I,J)*C(J)
2280 NEXT J
2290 Q(I,0)=X(I,0)-S
2300 NEXT I
2310 REM EV
2320 ESS=0
2330 FOR I=1 TO N
2340 ESS=ESS+Q(I,0)*Q(I,0)
2350 NEXT I
2360 EV=ESS/V1
2370 RETURN
2380 REM INVERT R
2390 FOR I=1 TO K
2400 V(I,I)=1/R(I,I)
2410 NEXT I
2420 FOR I=K-1 TO 1 STEP -1
2430 FOR J=I+1 TO K
2440 S=0
2450 FOR L=I+1 TO J
2460 S=S+R(I,L)*V(L,J)
2470 NEXT L
2480 V(I,J)=-S/R(I,I)
2490 NEXT J :: NEXT I
2500 RETURN
2510 REM UNSCALED MATRIX
2520 FOR I=1 TO K
2530 FOR J=1 TO K
2540 R(I,J)=0
2550 FOR L=1 TO K
2560 R(I,J)=R(I,J)+V(I,L)*V(J,L)
2570 NEXT L :: NEXT J :: NEXT I
2580 RETURN
2590 REM RESULTS
2600 REM EQUATION
2610 GOSUB 2650
2620 REM SUMMARY STATISTICS
2630 GOSUB 3150
2640 RETURN
2650 REM EQUATION
2660 CALL CLEAR
2670 CALL HCHAR(1,3,61,28)
2680 DISPLAY AT(2,6):"REGRESSION EQUATION"
2690 CALL HCHAR(3,3,61,28)
2700 DISPLAY AT(5,7):"ESTIMATED";TAB(21);"STANDARD
     "
2710 DISPLAY AT(6,1):"TERM";TAB(11);"VALUE";TAB(24
     );"ERROR"
2720 IMAGE ########.### ########.###
```

```
2730 ROW=8
2740 FOR I=1 TO K
2750 DISPLAY AT(ROW,1):B$(I-1)
2760 DISPLAY AT(ROW,4):USING 2720:B(I),SQR(EV*R(I,
     I))
2770 ROW=ROW+1 :: NEXT I
2780 CALL HCHAR(23,3,61,28)
2790 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
2800 ACCEPT AT(24,25):Z$
2810 RETURN
2820 REM SUMMARY TERMS
2830 REM SUMS OF SQUARES
2840 S=0 :: SS=0
2850 FOR I=1 TO N
2860 S=S+X(I,0)
2870 SS=SS+X(I,0)*X(I,0)
2880 NEXT I
2890 TSS=SS-S*S/N
2900 RSS=TSS-ESS
2910 REM GOODNESS-OF-FIT
2920 RSQ=RSS/TSS
2930 RBSQ=1-(ESS/V1)/(TSS/(N-1))
2940 F=RSS/(K-1)/EV
2950 REM DW STATISTIC
2960 S=0
2970 FOR I=2 TO N
2980 S=S+(Q(I,0)-Q(I-1,0))^2
2990 NEXT I
3000 DW=S/ESS
3010 REM RHO
3020 REM NUMERATOR
3030 S=0
3040 FOR I=2 TO N
3050 S=S+Q(I,0)*Q(I-1,0)
3060 NEXT I
3070 REM DENOMINATOR
3080 D=0
3090 FOR I=2 TO N-1
3100 D=D+Q(I,0)^2
3110 NEXT I
3120 REM QUOTIENT
3130 RHO=S/D
3140 RETURN
3150 REM SUMMARY STATISTICS
3160 CALL CLEAR
3170 CALL HCHAR(1,3,61,28)
3180 DISPLAY AT(2,6):"SUMMARY STATISTICS"
3190 CALL HCHAR(3,3,61,28)
3200 DISPLAY AT(4,22):"DEGREES" :: DISPLAY AT(5,14
     ):"SUMS OF";TAB(25);"OF"
```

```
3210 DISPLAY AT(6,14):"SQUARES";TAB(22);"FREEDOM"
3220 IMAGE ##########.###{3 SPACES}###
3230 DISPLAY AT(8,1):"TOTAL" :: DISPLAY AT(8,6):US
     ING 3220:TSS,N-1
3240 DISPLAY AT(9,1):"REG'N" :: DISPLAY AT(9,6):US
     ING 3220:RSS,K-1
3250 DISPLAY AT(10,1):"ERROR" :: DISPLAY AT(10,6):
     USING 3220:ESS,V1
3260 IMAGE =##########.###
3270 DISPLAY AT(12,1):"R-SQUARED" :: DISPLAY AT(12
     ,15):USING 3260:RSQ
3280 DISPLAY AT(13,1):"R BAR SQUARED" :: DISPLAY A
     T(13,15):USING 3260:RBSQ
3290 DISPLAY AT(15,1):"F STATISTIC" :: DISPLAY AT(
     15,15):USING 3260:F
3300 DISPLAY AT(17,1):"ERROR VARIANCE" :: DISPLAY
     AT(17,15):USING 3260:EV
3310 DISPLAY AT(19,1):"STANDARD ERROR"
3320 DISPLAY AT(20,2):"OF ESTIMATE" :: DISPLAY AT(
     20,15):USING 3260:SQR(EV)
3330 CALL HCHAR(23,3,61,28)
3340 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
3350 ACCEPT AT(24,25):Z$
3360 CALL CLEAR
3370 CALL HCHAR(1,3,61,28)
3380 DISPLAY AT(2,4):"ANALYSIS OF RESIDUALS"
3390 CALL HCHAR(3,3,61,28)
3400 DISPLAY AT(5,1):"CORRELATION"
3410 DISPLAY AT(6,2):"COEFFICIENT" :: DISPLAY AT(6
     ,15):USING 3260:RHO
3420 DISPLAY AT(8,1):"DURBIN-WATSON"
3430 DISPLAY AT(9,2):"STATISTIC" :: DISPLAY AT(9,1
     5):USING 3260:DW
3440 CALL HCHAR(23,3,61,28)
3450 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
3460 ACCEPT AT(24,25):Z$
3470 RETURN
```

t-Curve Critical Values

When using regression analysis, you seldom know for sure if a particular explanatory variable actually influences Y. You tentatively choose such variables on the basis of accepted theory, common sense, or intuition; then, using an operation called the t-test, you decide which explanatory variables are important and which are superfluous.

Returning to the example from multiple linear regression,

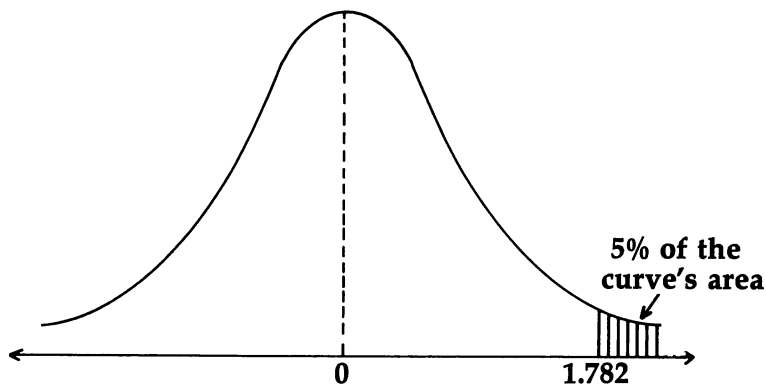
suppose you want to determine whether the number of automobile drivers in a state really does influence the yearly toll of traffic fatalities. The explanatory variable is the number of automobile drivers; to see if it actually affects the annual fatality toll, simply put it to the t-curve test.

To conduct the test, you have to do four things: create a null hypothesis, establish a test criterion, compute a t-value, and compare the computed t-value to the observed t-value. If the computed t-value is greater than the t-critical value, you can reject the null hypothesis and conclude that the explanatory variable really does influence the outcome.

In this case, the null hypothesis states that the number of fatalities is *not* related to the number of drivers. You will test it by computing an actual t-value, and you will reject it if that value is greater than the t-critical value at the 5 percent level of significance. That means that there is a 5 percent chance of rejecting the null hypothesis as false when it is actually true.

The t-critical value depends on the number of degrees of freedom for each particular case. How many degrees of freedom? To find out, simply subtract the number of parameters (3) from the number of observations on the regression equation (15). In this example, there are 12 degrees of freedom, and the computed t-critical value (which your TI generates when you key in 12) is 1.782.

Figure 4-6. t-Curve with 12 Degrees of Freedom



The t-critical value is 1.782. Five percent of all t-values are to the right of this point. If the actual t-ratio is greater than the critical value, we will reject the null hypothesis and conclude that fatalities do indeed increase when more people drive.

Knowing the computed t-value, you're ready to complete your analysis by finding the observed t-value. The observed t-value is equal to the estimated regression coefficient divided by its standard error. Since the regression coefficient for "number of drivers" is 3.778, and since its standard error is 0.432, the t-value is approximately 8.7. That is greater than the t-critical value, so you would reject the null hypothesis and conclude that fatalities do indeed increase when more people drive.

Although the t-test is never foolproof, it is useful for sorting the wheat from the chaff in regression analysis. To save the trouble of looking up t-critical values in the back of statistics books, the TI generates these numbers for you. It does this for so-called one-tail tests, in which the explanatory variable can influence the outcome in only one way, as well as for two-tail tests, where that influence can be either positive or negative.

Program 4-4. t-Curve Critical Values

```
100 REM t-CURVE CRITICAL VALUES
130 REM INITIALIZE
140 GOSUB 200
150 REM COMPUTE VALUES
160 GOSUB 520
170 REM DISPLAY VALUES
180 GOSUB 950
190 END
200 REM INITIALIZE
210 REM HEADING
220 GOSUB 260
230 REM ENTER DEGREES OF FREEDOM
240 GOSUB 450
250 RETURN
260 REM HEADING
270 CALL CLEAR
280 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES"
290 DISPLAY AT(2,1):"t-CURVE CRITICAL VALUES"
300 DISPLAY AT(3,1):"FOR USE IN HYPOTHESIS"
310 DISPLAY AT(4,1):"TESTING."
320 DISPLAY AT(6,1):"SIMPLY ENTER THE NUMBER"
330 DISPLAY AT(7,1):"OF DEGREES OF FREEDOM,"
340 DISPLAY AT(8,1):"AND CRITICAL VALUES"
350 DISPLAY AT(9,1):"WILL BE TALLIED AT THE"
360 DISPLAY AT(10,1):"10%, 5%, & 1% LEVELS"
370 DISPLAY AT(11,1):"OF SIGNIFICANCE, FOR A"
380 DISPLAY AT(12,1):"TWO-TAIL TEST."
```

```
390 DISPLAY AT(14,1):"AND THEY WILL BE TALLIED"
400 DISPLAY AT(15,1):"AT THE 5.0%, 2.5%, & 0.5%"
410 DISPLAY AT(16,1):"LEVELS FOR A ONE-TAIL TEST."
420 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
430 ACCEPT AT(23,25):Z$
440 RETURN
450 REM ENTER DEGREES OF FREEDOM
460 CALL CLEAR
470 DISPLAY AT(1,1):"HOW MANY DEGREES OF"
480 DISPLAY AT(2,1):"FREEDOM ARE THERE ?"
490 ACCEPT AT(2,20)BEEP VALIDATE(NUMERIC,""):V$
500 IF V$="" THEN 490 ELSE V=INT(VAL(V$))
510 RETURN
520 REM COMPUTE VALUES
530 CALL CLEAR
540 DISPLAY AT(1,1):"COMPUTING VALUES ..."
550 REM READ-IN VALUES FOR APPROXIMATION FORMULA
560 GOSUB 600
570 REM COMPUTE; USE ACTUAL VALUES FOR LESS THAN 4
     D.O.F.
580 IF V<4 THEN GOSUB 720 ELSE GOSUB 840
590 RETURN
600 REM READ-IN VALUES
610 DATA 1.6449,1.9600,2.5758
620 DATA 3.5283,0.60033,-0.82847
630 DATA 0.85602,0.95910,1.8745
640 DATA 1.2209,-0.90259,-2.2311
650 DATA -1.5162,0.11588,1.5631
660 READ A(1),A(2),A(3)
670 READ B(1),B(2),B(3)
680 READ C(1),C(2),C(3)
690 READ D(1),D(2),D(3)
700 READ E(1),E(2),E(3)
710 RETURN
720 REM ACTUAL VALUES
730 REM V=1
740 DATA 6.314,12.706,63.657
750 REM V=2
760 DATA 2.920,4.303,9.925
770 REM V=3
780 DATA 2.353,3.182,5.841
790 FOR I=1 TO 3
800 READ X,Y,Z
810 IF V=I THEN T(1)=X :: T(2)=Y :: T(3)=Z
820 NEXT I
830 RETURN
840 REM APPROXIMATION FORMULA
850 REM T(1)=10% LEVEL, T(2)=5% LEVEL, T(3)=1% LEV
    EL
```

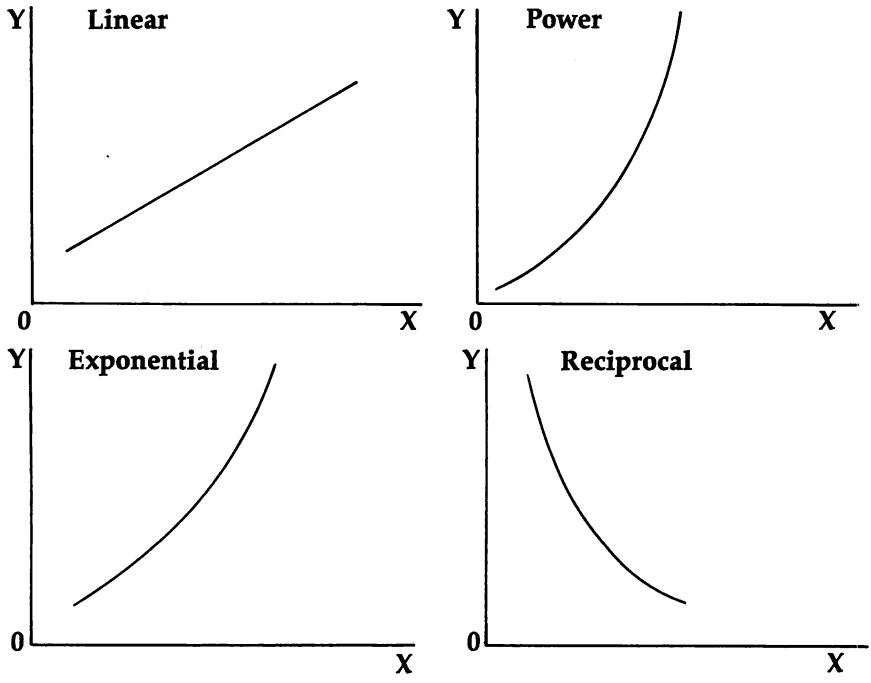
```
860 FOR I=1 TO 3
870 REM NUMERATOR
880 NU=A(I)*V+B(I)+C(I)/V
890 REM DENOMINATOR
900 DE=V+D(I)+E(I)/V
910 REM QUOTIENT
920 T(I)=NU/DE
930 NEXT I
940 RETURN
950 REM DISPLAY VALUES
960 REM TWO-TAIL
970 GOSUB 1010
980 REM ONE-TAIL
990 GOSUB 1220
1000 RETURN
1010 REM TWO-TAIL
1020 CALL CLEAR
1030 CALL HCHAR(1,3,61,28)
1040 DISPLAY AT(2,3):"t-CURVE CRITICAL VALUES"
1050 DISPLAY AT(3,5):"FOR A TWO-TAIL TEST"
1060 CALL HCHAR(4,3,61,28)
1070 DISPLAY AT(6,5):"LEVEL";TAB(14);"CRITICAL VAL
     UES"
1080 DISPLAY AT(7,6):"OF"
1090 DISPLAY AT(8,1):"SIGNIFICANCE";TAB(16);"LOWER
     ";TAB(24);"UPPER"
1100 L$(1)="10%" :: L$(2)=" 5%" :: L$(3)=" 1%"
1110 IMAGE ####.### ###.###
1120 ROW=10
1130 FOR I=1 TO 3
1140 DISPLAY AT(ROW,5):L$(I)
1150 DISPLAY AT(ROW,13):USING 1110:-T(I),T(I)
1160 ROW=ROW+2
1170 NEXT I
1180 CALL HCHAR(23,3,61,28)
1190 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1200 ACCEPT AT(24,25):Z$
1210 RETURN
1220 REM ONE-TAIL VALUES
1230 CALL CLEAR
1240 CALL HCHAR(1,3,61,28)
1250 DISPLAY AT(2,3):"t-CURVE CRITICAL VALUES"
1260 DISPLAY AT(3,5):"FOR A ONE-TAIL TEST"
1270 CALL HCHAR(4,3,61,28)
1280 DISPLAY AT(6,5):"LEVEL";TAB(17);"CRITICAL"
1290 DISPLAY AT(7,6):"OF";TAB(18);"VALUE"
1300 DISPLAY AT(8,1):"SIGNIFICANCE";TAB(18);"+ OR
     -"
1310 L$(1)="5.0%" :: L$(2)="2.5%" :: L$(3)="0.5%"
```

```
1320 IMAGE ####.###
1330 ROW=10
1340 FOR I=1 TO 3
1350 DISPLAY AT(ROW,5):L$(I)
1360 DISPLAY AT(ROW,15):USING 1320:T(I)
1370 ROW=ROW+2
1380 NEXT I
1390 CALL HCHAR(23,3,61,28)
1400 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1410 ACCEPT AT(24,25):Z$
1420 RETURN
```

General-Form Curve Fitter

The relationship between one variable and another is not always linear. To handle nonlinear cases, you can use the General-Form Curve Fitter routine. It lets you estimate a simple regression equation using linear, power, exponential, or reciprocal forms, as shown in Figure 4-7.

Fgure 4-7. Relationships Between Two Variables



Suppose, for example, that you want to examine the relationship between sales of a popular microcomputer and time, using the observations shown in Table 4-4 and plotted in Figure 4-8. Since the graph reveals that sales have been climbing exponentially, you don't want to use the traditional linear equation. Instruct the TI to use the exponential one instead.

After entering the data, letting Y equal sales and X equal time (1976 = 1, 1977 = 2, and so on), the TI estimates the equation and displays these regression results:

REGRESSION RESULTS

FOR Y = A * (E) ↑ (B*X)

TERM	ESTIMATED VALUE	T-RATIO
A	0.198	-5.817
B	1.093	19.835
R-SQUARED	=	0.985
F STATISTIC	=	393.436
STANDARD ERROR OF THE ESTIMATE	=	0.357

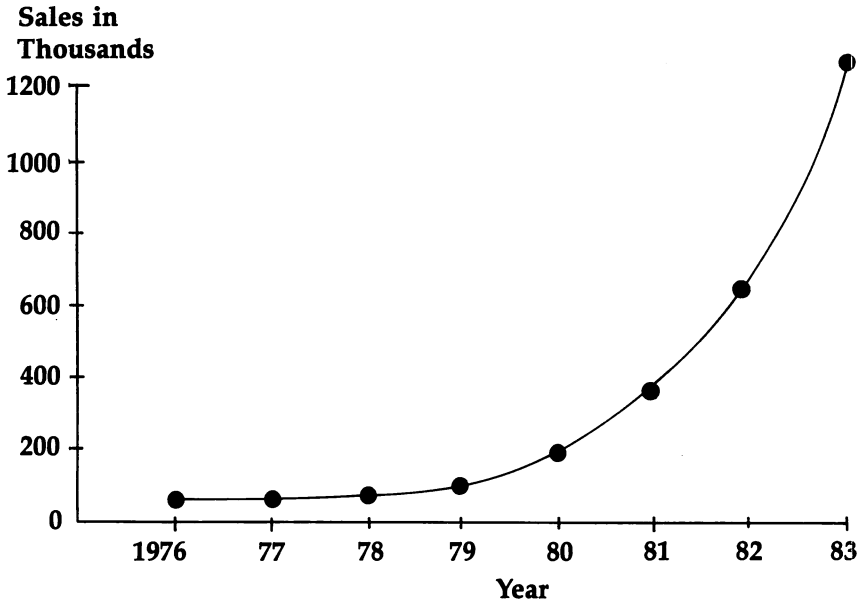
All these statistics are similar to ones you've seen before, so there's no need to go over them again. Note, however, the high value of the R-squared term. That means that the exponential equation fits the data very well, explaining 98.5 percent of the variation in sales.

Finally, if you wish, you can instruct the TI to estimate another regression equation on the same set of data, perhaps using a power function.

Table 4-4. Computer Sales

Year	Units Sold in Thousands	Time
1976	1.0	1
1977	1.5	2
1978	4.0	3
1979	10.0	4
1980	40.5	5
1981	200.0	6
1982	500.0	7
1983	1,200.0	8

Figure 4-8. Growth in Sales



Notes on the Program

General-Form Curve Fitter estimates any of the following equations:

$Y = a + bX$	Linear
$Y = aX^b$	Power
$Y = a(e)^{bX}$	Exponential
$Y = a + b/X$	Reciprocal

It does this using the *linear* least-squares algorithm. The trick, of course, is transforming the power, exponential, and reciprocal functions into linear form.

In the first two cases, this is done simply by taking the logarithm (to base e) of each side of the equation:

$\ln(Y) = \ln(a) + b*\ln(X)$	Power
$\ln(Y) = \ln(a) + bX$	Exponential

However, logarithms don't exist for negative numbers or for zero. If the TI encounters either of these conditions, it will return you to the equation-selection part of the program after letting you know what the problem was.

Further, when the power and exponential equations are computed, the estimated value of *a* (rather than ln(a)) is displayed. That is why the mathematical signs of *a* and its corresponding t-ratio will sometimes differ (the t-ratio is based on ln(a), as statistical theory demands).

Finally, when the reciprocal equation is estimated, values of X are simply divided into 1 to get the transformed variable. Division by zero is checked for and avoided.

Just as Don Quixote transformed windmills into "lawless giants" so he could do battle, so you can take on nonlinear equations by turning them into linear form. Don Quixote didn't know the difference, and neither does the TI.

Program 4-5. General-Form Curve Fitter

```
100 REM GENERAL-FORM CURVE FITTER
130 REM INITIALIZE
140 GOSUB 280
150 REM ENTER & EDIT DATA
160 GOSUB 680
170 REM SELECT FUNCTIONAL FORM
180 GOSUB 1400
190 IF CHOICE=5 THEN 270
200 REM TRANSFORM DATA
210 GOSUB 1510
220 REM ESTIMATE EQUATION
230 IF D$="BAD DATA" THEN 180 ELSE GOSUB 2000
240 REM DISPLAY RESULTS
250 GOSUB 2380
260 GOTO 180
270 END
280 REM INITIALIZE
290 REM HEADING
300 GOSUB 340
310 REM MAXIMUM NUMBER OF OBSERVATIONS
320 GOSUB 520
330 RETURN
340 REM HEADING
350 CALL CLEAR
360 DISPLAY AT(1,1):"THIS PROGRAM ESTIMATES A"
370 DISPLAY AT(2,1):"SIMPLE REGRESSION EQUATION"
380 DISPLAY AT(3,1):"USING ANY OF THE FOLLOWING"
390 DISPLAY AT(4,1):"FUNCTIONAL FORMS:"
400 GOSUB 440
410 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
420 ACCEPT AT(24,25):Z$
430 RETURN
```

```
440 REM FUNCTIONAL FORMS
450 DISPLAY AT(7,3):"EQUATION";TAB(20);"FORM"
460 DISPLAY AT(10,1):"1. Y = A + B*X";TAB(18);"LIN
    EAR"
470 DISPLAY AT(12,1):"2. Y = A*X^B";TAB(18);"POWER
    "
480 DISPLAY AT(14,1):"3. Y = A*e^(B*X)";TAB(18);"E
    XPONENTIAL"
490 DISPLAY AT(16,1):"4. Y = A + B/X";TAB(18);"REC
    IPROCAL"
500 DISPLAY AT(18,1):"5.";TAB(18);"NONE"
510 RETURN
520 REM MAXIMUM NUMBER OF OBSERVATIONS
530 CALL CLEAR
540 OPTION BASE 1
550 DATA 100
560 READ MAXN
570 DIM X(100,2),XT(100),YT(100)
580 V$(1)="Y" :: V$(2)="X"
590 DISPLAY AT(1,1):"THE MAXIMUM NUMBER OF"
600 DISPLAY AT(2,1):"OBSERVATIONS ALLOWED"
610 DISPLAY AT(3,1):"IS";MAXN;"."
620 DISPLAY AT(6,1):"FOR A DIFFERENT LIMIT,"
630 DISPLAY AT(7,1):"STOP PROGRAM EXECUTION AND"
640 DISPLAY AT(8,1):"CHANGE LINES 550 & 570."
650 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
660 ACCEPT AT(24,25):Z$
670 RETURN
680 REM ENTER & EDIT DATA
690 REM ENTER
700 GOSUB 740
710 REM EDIT
720 GOSUB 1060
730 RETURN
740 REM ENTER
750 REM ON Y
760 GOSUB 800
770 REM ON X
780 GOSUB 950
790 RETURN
800 REM ON Y
810 CALL CLEAR
820 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
830 DISPLAY AT(2,1):"DEPENDENT VARIABLE (Y)."
840 DISPLAY AT(3,1):"HIT 'ENTER' WHEN THROUGH."
850 N=MAXN
860 FOR I=1 TO MAXN
870 DISPLAY AT(8,1):"Y(";I;
880 DISPLAY AT(8,8):")= ?"
```

```
890 ACCEPT AT(8,13)VALIDATE(NUMERIC,""):S$
900 IF S$="" THEN N=I-1 :: I=MAXN :: GOTO 920
910 X(I,1)=VAL(S$)
920 NEXT I
930 IF N<3 THEN DISPLAY AT(23,1):"AT LEAST 3 OBSER
    VATIONS ARE" :: DISPLAY AT(24,1):"NEEDED. PLEA
    SE TRY AGAIN." :: GOTO 850
940 RETURN
950 REM ENTER DATA ON X
960 CALL CLEAR
970 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE"
980 DISPLAY AT(2,1):"EXPLANATORY VARIABLE (X)."
990 FOR I=1 TO N
1000 DISPLAY AT(7,1):"X(";I;
1010 DISPLAY AT(7,8):")= ?"
1020 ACCEPT AT(7,13)VALIDATE(NUMERIC,""):S$
1030 IF S$="" THEN X(I,2)=0 ELSE X(I,2)=VAL(S$)
1040 NEXT I
1050 RETURN
1060 REM EDIT DATA
1070 FOR I=1 TO 2
1080 FOR L=0 TO INT((N-1)/10)
1090 REM DISPLAY DATA
1100 GOSUB 1160
1110 REM CORRECT DATA
1120 GOSUB 1240
1130 NEXT L
1140 NEXT I
1150 RETURN
1160 REM DISPLAY DATA
1170 CALL CLEAR
1180 DISPLAY AT(1,1):"THESE ARE VALUES OF ";V$(I);
     ":"
1190 FOR J=1 TO 10
1200 Q=J+L*10
1210 IF Q<=N THEN DISPLAY AT(J+2,1):V$(I);"(";Q;TA
     B(8);")= ";X(Q,I)
1220 NEXT J
1230 RETURN
1240 REM CORRECT DATA
1250 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1260 ACCEPT AT(16,20)VALIDATE("YN","")BEEP:S$
1270 IF S$="N" THEN 1390
1280 IF S$="" THEN 1260
1290 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1300 DISPLAY AT(19,1):"DATUM TO BE CORRECTED ?"
1310 ACCEPT AT(19,24)BEEP VALIDATE(NUMERIC,""):Q$
1320 IF Q$="" THEN 1310 ELSE Q=INT(VAL(Q$))
```

```
1330 IF Q<(1+L*10)OR Q>N OR Q>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER NUMBER WITHIN" :: DI
     SPLAY AT(22,1):"BOUNDS SHOWN." :: GOTO 1290
1340 DISPLAY AT(21,1):"WHAT SHOULD THE"
1350 DISPLAY AT(22,1):"VALUE BE ?"
1360 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1370 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
1380 GOSUB 1170 :: GOTO 1250
1390 RETURN
1400 REM SELECT FUNCTIONAL FORM
1410 CALL CLEAR
1420 DISPLAY AT(1,1):"PLEASE ENTER THE TYPE OF"
1430 DISPLAY AT(2,1):"EQUATION THAT YOU WANT TO"
1440 DISPLAY AT(3,1):"ESTIMATE."
1450 GOSUB 440
1460 DISPLAY AT(20,1):"NUMBER (1 TO 5) = ?"
1470 ACCEPT AT(20,20)BEEP VALIDATE(DIGIT,""):CHOIC
     E$
1480 IF CHOICE$="" THEN 1470 ELSE CHOICE=VAL(CHOIC
     E$)
1490 IF CHOICE<1 OR CHOICE>5 THEN DISPLAY AT(23,1)
     :"PLEASE ENTER A NUMBER" :: DISPLAY AT(24,1):
     "FROM 1 TO 5" :: GOTO 1470
1500 RETURN
1510 REM TRANSFORM DATA
1520 CALL CLEAR
1530 DISPLAY AT(1,1):"TRANSFORMING DATA ..."
1540 D$="GOOD DATA"
1550 ON CHOICE GOSUB 1570,1620,1780,1840
1560 RETURN
1570 REM LINEAR FUNCTION
1580 FOR I=1 TO N
1590 YT(I)=X(I,1):: XT(I)=X(I,2)
1600 NEXT I
1610 RETURN
1620 REM POWER FUNCTION
1630 FOR I=1 TO N
1640 IF X(I,1)>0 THEN YT(I)=LOG(X(I,1))ELSE SYMBOL
     $="Y" :: INDEX=I :: VALUE=X(I,1):: GOSUB 1680
     :: I=N :: GOTO 1660
1650 IF X(I,2)>0 THEN XT(I)=LOG(X(I,2))ELSE SYMBOL
     $="X" :: INDEX=I :: VALUE=X(I,2):: GOSUB 1680
     :: I=N
1660 NEXT I
1670 RETURN
1680 REM BAD DATA
1690 CALL CLEAR
1700 DISPLAY AT(1,1):"SORRY, BUT I CAN'T ESTIMATE"
1710 DISPLAY AT(2,1):"YOUR EQUATION."
```

```
1720 DISPLAY AT(4,1):SYMBOL$;"(";INDEX;") =";VALUE
1730 DISPLAY AT(5,1):"AND I CAN'T TAKE THE LOG."
1740 D$="BAD DATA"
1750 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1760 ACCEPT AT(24,25):Z$
1770 RETURN
1780 REM EXPONENTIAL FUNCTION
1790 FOR I=1 TO N
1800 XT(I)=X(I,2)
1810 IF X(I,1)>0 THEN YT(I)=LOG(X(I,1))ELSE SYMBOL
     $="Y" :: INDEX=I :: VALUE=X(I,1):: GOSUB 1680
     :: I=N
1820 NEXT I
1830 RETURN
1840 REM RECIPROCAL FUNCTION
1850 FOR I=1 TO N
1860 YT(I)=X(I,1)
1870 IF X(I,2)<>0 THEN XT(I)=1/X(I,2)ELSE INDEX=I
     :: GOSUB 1900 :: I=N
1880 NEXT I
1890 RETURN
1900 REM DIVISION BY ZERO
1910 CALL CLEAR
1920 DISPLAY AT(1,1):"SORRY, BUT I CAN'T ESTIMATE"
1930 DISPLAY AT(2,1):"YOUR EQUATION."
1940 DISPLAY AT(4,1):"X(";INDEX;") IS 0, AND I"
1950 DISPLAY AT(5,1):"CAN'T DIVIDE."
1960 D$="BAD DATA"
1970 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
1980 ACCEPT AT(24,25):Z$
1990 RETURN
2000 REM ESTIMATE EQUATION
2010 CALL CLEAR
2020 DISPLAY AT(1,1):"ESTIMATING EQUATION ..."
2030 REM A & B
2040 GOSUB 2120
2050 REM ANOVA TERMS
2060 GOSUB 2240
2070 REM T-STATISTICS
2080 GOSUB 2320
2090 REM TRANSFORM A & B BACK TO NON-LOG FORM
2100 IF CHOICE=2 OR CHOICE=3 THEN A=EXP(A)
2110 RETURN
2120 REM A & B
2130 SX=0 :: SY=0 :: XQ=0 :: YQ=0 :: CP=0
2140 FOR I=1 TO N
2150 SX=SX+XT(I)
2160 SY=SY+YT(I)
2170 XQ=XQ+XT(I)*XT(I)
```

```
2180 YQ=YQ+YT(I)*YT(I)
2190 CP=CP+XT(I)*YT(I)
2200 NEXT I
2210 B=(N*CP-SX*SY)/(N*XQ-SX*SX)
2220 A=(SY-B*SX)/N
2230 RETURN
2240 REM ANOVA TERMS
2250 TSS=YQ-SY*SY/N
2260 RSS=B*(CP-SX*SY/N)
2270 ESS=TSS-RSS
2280 RSQ=RSS/TSS
2290 EV=ESS/(N-2)
2300 F=RSS/EV
2310 RETURN
2320 REM T STATISTICS
2330 SB=SQR(EV/(XQ-SX*SX/N))
2340 SA=SQR(EV*XQ/(N*XQ-SX*SX))
2350 TB=B/SB
2360 TA=A/SA
2370 RETURN
2380 REM DISPLAY RESULTS
2390 CALL CLEAR
2400 CALL HCHAR(1,3,61,28)
2410 DISPLAY AT(2,6):"REGRESSION RESULTS"
2420 CALL HCHAR(3,3,61,28)
2430 IF CHOICE=1 THEN S$="A + B*X"
2440 IF CHOICE=2 THEN S$="A*X^B"
2450 IF CHOICE=3 THEN S$="A*e^(B*X)"
2460 IF CHOICE=4 THEN S$="A + B/X"
2470 DISPLAY AT(5,1):"FOR  Y = ";TAB(10);S$;" :"
2480 DISPLAY AT(7,8):"ESTIMATED";TAB(26);"t"
2490 DISPLAY AT(8,1):"TERM";TAB(10);"VALUE";TAB(24
     );"RATIO"
2500 IMAGE ##########.### #######.###
2510 IMAGE = #######.###
2520 DISPLAY AT(10,2):"A" :: DISPLAY AT(10,3):USIN
     G 2500:A,TA
2530 DISPLAY AT(11,2):"B" :: DISPLAY AT(11,3):USIN
     G 2500:B,TB
2540 DISPLAY AT(15,1):"R-SQUARED" :: DISPLAY AT(15
     ,16):USING 2510:RSQ
2550 DISPLAY AT(17,1):"F STATISTIC" :: DISPLAY AT(
     17,16):USING 2510:F
2560 DISPLAY AT(19,1):"STANDARD ERROR"
2570 DISPLAY AT(20,1):"OF THE ESTIMATE" :: DISPLAY
      AT(20,16):USING 2510:SQR(EV)
2580 CALL HCHAR(23,3,61,28)
2590 DISPLAY AT(24,1):"HIT 'ENTER' TO CONTINUE"
2600 ACCEPT AT(24,25):Z$
2610 RETURN
```

Chapter 5

Matrix Manipulations

Matrix Manipulations

Matrices are rectangular arrays of numbers or variables. They represent unwieldy arithmetic relationships or large groups of data in a relatively simple, easy-to-understand fashion.

Because of their great utility, matrices are employed in science, in business, and in many computer programs. This chapter presents several common matrix routines that should prove useful:

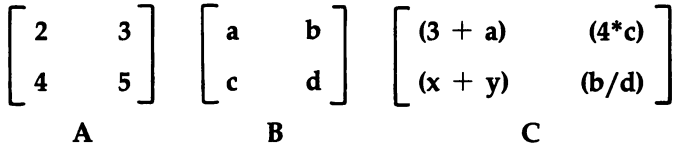
- "Matrix Addition and Subtraction"
- "Matrix Multiplication"
- "Sweet and Simple Matrix Inversion"
- "Determinant of a Matrix"
- "Matrix Inversion Using Gauss-Jordan Sweep with Complete Pivoting"

You can use these routines as they stand, or as subroutines in larger programs.

Matrix Addition and Subtraction

Matrices are groups of numbers or variables, arranged in rectangular fashion and enclosed in brackets. The items in a matrix are called its elements. As Figure 5-1 illustrates, these can be single numbers, variables, or expressions.

Figure 5-1. Examples of Matrices



The elements of matrices A, B, and C are numbers, variables, and expressions, respectively.

Matrices are described in terms of their dimensions. A matrix with only one dimension is called a row or column vector, while a matrix of two dimensions generally contains several rows and columns. A matrix may also have three dimensions, containing rows, columns, and layers.

In specifying the size of a two-dimensional array, the expression R x C is used. R is the number of rows in the matrix, and C is the number of columns. A matrix with three rows and seven columns is of order, or size, 3 x 7. Similarly, a matrix with ten rows and eight columns is of order 10 x 8.

Matrices can be added or subtracted only if they are the same size. When they are, the matrices are said to be *conformable* for addition or subtraction. The sum of two matrices is the sum of the corresponding elements of each, while the difference is the difference between corresponding elements. Figure 5-2 illustrates the mechanics of matrix addition and subtraction.

Figure 5-2. Matrix Addition and Subtraction

Addition:

$$\begin{bmatrix} 5 & 3 \\ 7 & 0 \end{bmatrix} + \begin{bmatrix} 2 & 4 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} (5+2) & (3+4) \\ (7+1) & (0+3) \end{bmatrix} = \begin{bmatrix} 7 & 7 \\ 8 & 3 \end{bmatrix}$$

A B C

In general, C(I,J) = A(I,J) + B(I,J), where I stands for the Ith row of a matrix, and where J stands for the Jth column.

Subtraction:

$$\begin{bmatrix} 5 & 3 \\ 7 & 0 \end{bmatrix} - \begin{bmatrix} 2 & 4 \\ 1 & 3 \end{bmatrix} + \begin{bmatrix} (5-2) & (3-4) \\ (7-1) & (0-3) \end{bmatrix} + \begin{bmatrix} 3 & -1 \\ 6 & -3 \end{bmatrix}$$

In using the TI computer program for adding and subtracting two matrices, first type in the elements of each. The program allows you to review and edit your entries. You then tell the computer to add or to subtract arrays. In the latter case, the second matrix is always subtracted from the first.

Program 5-1. Matrix Addition and Subtraction

```
100 REM MATRIX ADDITION & SUBTRACTION
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
```

```
160 GOSUB 690
170 REM EDIT DATA
180 GOSUB 840
190 REM DISPLAY RESULT
200 GOSUB 1220
210 END
220 REM INITIALIZE
230 REM HEADING
240 GOSUB 280
250 REM ENTER ORDER
260 GOSUB 450
270 RETURN
280 REM HEADING
290 CALL CLEAR
300 DISPLAY AT(1,1):"THIS PROGRAM ADDS OR"
310 DISPLAY AT(2,1):"SUBTRACTS ANY TWO MATRICES."
320 REM MAX SIZE
330 DATA 15,20
340 DIM X(2,15,20)
350 READ MROW,MCOL
360 DISPLAY AT(4,1):"THE MAXIMUM ALLOWABLE SIZE"
370 DISPLAY AT(5,1):"OF EACH MATRIX IS:"
380 DISPLAY AT(7,3):"ROWS = ";MROW
390 DISPLAY AT(8,3):"COLS = ";MCOL
400 DISPLAY AT(10,1):"CHANGE LINES 330 & 340"
410 DISPLAY AT(11,1):"FOR DIFFERENT VALUES."
420 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
430 ACCEPT AT(23,25):Z$
440 RETURN
450 REM ENTFR ORDER
460 REM ROWS
470 GOSUB 510
480 REM COLUMNS
490 GOSUB 600
500 RETURN
510 REM ROWS
520 CALL CLEAR
530 DISPLAY AT(1,1):"HOW MANY ROWS DO YOUR"
540 DISPLAY AT(2,1):"MATRICES HAVE ?"
550 ACCEPT AT(2,16)BEEP VALIDATE(DIGIT,""):S$
560 IF S$="" THEN 550 ELSE NROW=VAL(S$)
570 IF NROW=0 THEN DISPLAY AT(23,1):"AT LEAST 1 RO
    W IS NEEDED" :: GOTO 550
580 IF NROW>MROW THEN DISPLAY AT(23,1):"ONLY";MROW
    ;"ROWS ALLOWED" :: GOTO 550
590 RETURN
600 REM COLUMNS
610 CALL CLEAR
620 DISPLAY AT(1,1):"HOW MANY COLUMNS DO YOUR"
```

```
630 DISPLAY AT(2,1):"MATRICES HAVE ?"
640 ACCEPT AT(2,16)BEEP VALIDATE(DIGIT,""):S$
650 IF S$="" THEN 640 ELSE NCOL=VAL(S$)
660 IF NCOL=0 THEN DISPLAY AT(23,1):"AT LEAST 1 CO
    LUMN IS NEEDED" :: GOTO 640
670 IF NCOL>MCOL THEN DISPLAY AT(23,1):"ONLY";MCOL
    ;"COLUMNS ALLOWED" :: GOTO 640
680 RETURN
690 REM ENTER DATA
700 FOR Q=1 TO 2
710 IF Q=1 THEN T$=" 1ST" ELSE T$=" 2ND"
720 FOR I=1 TO NCOL
730 CALL CLEAR
740 DISPLAY AT(1,1):"PLEASE ENTER DATA ON THE";T$
750 DISPLAY AT(2,1):"MATRIX FOR COLUMN #";I;"."
760 FOR J=1 TO NROW
770 DISPLAY AT(4,1):"ROW #";J;TAB(10);"= ? "
780 ACCEPT AT(4,14)VALIDATE(NUMERIC,""):X$
790 IF X$="" THEN X(Q,J,I)=0 ELSE X(Q,J,I)=VAL(X$)
800 NEXT J
810 NEXT I
820 NEXT Q
830 RETURN
840 REM EDIT DATA
850 FOR Z=1 TO 2
860 IF Z=1 THEN T$=" 1ST" ELSE T$=" 2ND"
870 FOR I=1 TO NCOL
880 FOR L=0 TO INT((NROW-1)/10)
890 REM DISPLAY DATA
900 GOSUB 970
910 REM CORRECT DATA
920 GOSUB 1060
930 NEXT L
940 NEXT I
950 NEXT Z
960 RETURN
970 REM DISPLAY DATA
980 CALL CLEAR
990 DISPLAY AT(1,1):"THESE ARE VALUES OF THE";T$
1000 DISPLAY AT(2,1):"MATRIX FOR COLUMN";I;":"
1010 FOR J=1 TO 10
1020 M=J+L*10
1030 IF M<=NROW THEN DISPLAY AT(J+3,1):"ROW(";M;TA
     B(9);")= ";X(Z,M,I)
1040 NEXT J
1050 RETURN
1060 REM CORRECT DATA
1070 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1080 ACCEPT AT(16,20)VALIDATE("YN","")SIZE(1)BEEP:S$
```

```
1090 IF S$="N" THEN 1210
1100 IF S$="" THEN 1080
1110 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1120 DISPLAY AT(19,1):"ROW TO BE CORRECTED ?"
1130 ACCEPT AT(19,23)BEEP VALIDATE(DIGIT,""):Q$
1140 IF Q$="" THEN 1130 ELSE Q=VAL(Q$)
1150 IF Q<(1+L*10)OR Q>NROW OR Q>(10+L*10)THEN DIS
     PLAY AT(21,1):"PLEASE ENTER ROW WITHIN" :: DI
     SPLAY AT(22,1):"BOUNDS SHOWN." :: GOTO 1130
1160 DISPLAY AT(21,1):"WHAT SHOULD THE"
1170 DISPLAY AT(22,1):"VALUE BE ?"
1180 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1190 IF S$="" THEN X(Z,Q,I)=0 ELSE X(Z,Q,I)=VAL(S$
     )
1200 GOSUB 980 :: GOTO 1070
1210 RETURN
1220 REM DISPLAY RESULT
1230 REM ADD OR SUBTRACT
1240 GOSUB 1280
1250 REM DISPLAY
1260 GOSUB 1370
1270 RETURN
1280 REM + OR -
1290 CALL CLEAR
1300 DISPLAY AT(1,1):"WOULD YOU LIKE TO:"
1310 DISPLAY AT(3,1):"1. ADD"
1320 DISPLAY AT(4,1):"2. SUBTRACT"
1330 DISPLAY AT(6,1):"YOUR MATRICES (1 OR 2) ?"
1340 ACCEPT AT(6,25)BEEP VALIDATE("1 2","")SIZE(1)
     :S$
1350 IF S$="" THEN 1340 ELSE VALUE=VAL(S$)
1360 RETURN
1370 REM RESULTS
1380 REM IN 10 BY 2 BLOCKS
1390 FOR Q=1 TO NROW STEP 10
1400 GOSUB 1430
1410 NEXT Q
1420 RETURN
1430 REM PRINT
1440 IMAGE ########.###
1450 FOR I=1 TO NCOL STEP 2
1460 CALL CLEAR
1470 CALL HCHAR(1,3,61,28)
1480 IF VALUE=1 THEN DISPLAY AT(2,9):"MATRIX SUM:"
      ELSE DISPLAY AT(2,6):"MATRIX DIFFERENCE:"
1490 CALL HCHAR(3,3,61,28)
1500 REM PRINT COLUMN HEADING
1510 ROW=8 :: COL=7
1520 FOR L=I TO I+1
```

```
153Ø IF L<=NCOL THEN DISPLAY AT(5,COL):"COLUMN";L
154Ø COL=COL+13
155Ø NEXT L
156Ø REM VALUES
157Ø FOR J=Q TO Q+9
158Ø IF J>NROW THEN 168Ø
159Ø DISPLAY AT(ROW,1):"R"&STR$(J)
16ØØ COL=4
161Ø FOR L=I TO I+1
162Ø IF L<=NCOL AND VALUE=1 THEN N=X(1,J,L)+X(2,J,
     L)
163Ø IF L<=NCOL AND VALUE=2 THEN N=X(1,J,L)-X(2,J,
     L)
164Ø IF L<=NCOL THEN DISPLAY AT(ROW,COL):USING 144
     Ø:N
165Ø COL=COL+13
166Ø NEXT L
167Ø ROW=ROW+1
168Ø NEXT J
169Ø CALL HCHAR(23,3,61,28)
17ØØ DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
171Ø ACCEPT AT(24,26):Z$
172Ø NEXT I
173Ø RETURN
```

Matrix Multiplication

The job of multiplying two matrices, X and Y, is a bit more complex than the task of adding or subtracting them. First you must determine if the matrices can in fact be multiplied. In the expression X*Y, array X is called the *lead* matrix and array Y the *lag* matrix. X is postmultiplied by Y, or put another way, Y is premultiplied by X. A matrix product exists only if the number of columns in the lead matrix equals the number of rows in the lag matrix.

Suppose X is a matrix of order 3 x 4 (three rows and four columns), for example, and that Y is a matrix of order 4 x 7. According to the rule, the product X*Y exists since the number of columns in the former equals the number of rows in the latter. Note, however, that Y*X is undefined since the seven columns of the lead matrix do not jibe with the three rows of the lag matrix.

A handy way to apply our test is to jot down the dimensions of each array, side by side:

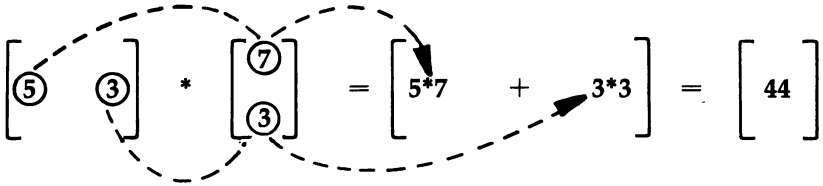
X*Y
(3x4)(4x7)

Multiplication is possible only when the two middle dimensions, 4's in this case, are equal.

But how do you actually compute the product of X and Y? By summing the products of row elements of matrix X and corresponding column elements of matrix Y, for all rows and columns. As Figure 5-3 illustrates, this is a lot easier than it sounds. But note that matrix multiplication differs from numerical multiplication in one important way: The matrix product X*Y will not usually equal the matrix product Y*X, even when both expressions are defined. Figure 5-4 gives an example.

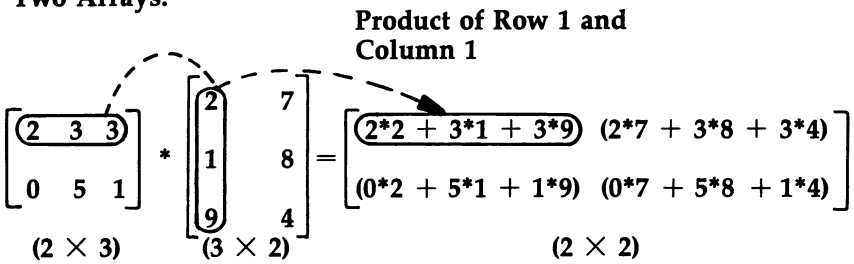
Figure 5-3. Examples of Matrix Multiplication

Two Vectors:



Notice here that the middle dimensions (the 2's) cancel out, giving us a 1 x 1 matrix.

Two Arrays:



Again, the middle dimensions (the 3's) cancel out, giving us a 2 x 2 matrix.

To use the TI matrix-multiplication routine, first enter the dimensions of the two arrays, X and Y. The TI checks the sizes of each to make sure that multiplication is in fact possible. Then enter, review, and edit your data, and the matrix product X*Y will be computed and displayed.

Figure 5-4. Another Example

$$\underset{X}{\begin{bmatrix} 2 & 3 \\ 0 & 4 \end{bmatrix}} * \underset{Y}{\begin{bmatrix} 1 & 0 \\ 2 & 7 \end{bmatrix}} = \begin{bmatrix} (2*1 + 3*2) & (2*0 + 3*7) \\ (0*1 + 4*2) & (0*0 + 4*7) \end{bmatrix}$$

$$= \begin{bmatrix} 8 & 21 \\ 8 & 28 \end{bmatrix}$$

Note, however, that

$$\underset{Y}{\begin{bmatrix} 1 & 0 \\ 2 & 7 \end{bmatrix}} * \underset{X}{\begin{bmatrix} 2 & 3 \\ 0 & 4 \end{bmatrix}} = \begin{bmatrix} 2 & 3 \\ 4 & 34 \end{bmatrix}$$

Hence, the matrix product X*Y does not equal the matrix product Y*X.

Program 5-2. Matrix Multiplication

```
100 REM MATRIX MULTIPLICATION
130 REM INITIALIZE
140 GOSUB 240
150 REM ENTER DATA
160 GOSUB 710
170 REM EDIT DATA
180 GOSUB 750
190 REM COMPUTE
200 GOSUB 790
210 REM DISPLAY RESULT
220 GOSUB 910
230 END
240 REM INITIALIZE
250 REM HEADING
260 GOSUB 300
270 REM ENTER MATRIX SIZES
280 GOSUB 520
290 RETURN
300 REM HEADING
310 CALL CLEAR
320 DISPLAY AT(1,1):"THIS PROGRAM MULTIPLIES TWO"
330 DISPLAY AT(2,1):"MATRICES, X & Y."
```

```
340 DISPLAY AT(4,1):"SPECIFICALLY, X IS POST-"
350 DISPLAY AT(5,1):"MULTIPLIED BY Y."
360 REM MAX SIZES
370 DATA 10,15
380 DATA 15,10
390 DIM X(10,15),Y(15,10),Z(15,15)
400 READ XROW,XCOL
410 READ YROW,YCOL
420 DISPLAY AT(7,1):"THE MAXIMUM ALLOWABLE SIZES"
430 DISPLAY AT(8,1):"OF THESE ARRAYS ARE:"
440 DISPLAY AT(10,4):"MATRIX X";TAB(19);"MATRIX Y"
450 DISPLAY AT(12,3):"ROWS =";XROW;TAB(18);"ROWS =
    ";YROW
460 DISPLAY AT(13,3):"COLS =";XCOL;TAB(18);"COLS =
    ";YCOL
470 DISPLAY AT(18,1):"CHANGE LINES 370 - 390"
480 DISPLAY AT(19,1):"FOR DIFFERENT VALUES."
490 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
500 ACCEPT AT(23,25):Z$
510 RETURN
520 REM MATRIX SIZES
530 CALL SZE("ROWS","X",RX,XROW)
540 CALL SZE("COLUMNS","X",CX,XCOL)
550 CALL SZE("ROWS","Y",RY,YROW)
560 CALL SZE("COLUMNS","Y",CY,YCOL)
570 REM CHECK FOR CONFORMABILITY
580 GOSUB 600
590 RETURN
600 REM CHECK
610 IF CX=RY THEN 700
620 CALL CLEAR
630 DISPLAY AT(5,1):"SORRY, BUT I CAN'T MULTIPLY"
640 DISPLAY AT(6,1):"YOUR MATRICES."
650 DISPLAY AT(8,1):"THE NUMBER OF COLUMNS IN"
660 DISPLAY AT(9,1):"MATRIX X (";CX;") DOES NOT"
670 DISPLAY AT(10,1):"EQUAL THE NUMBER OF ROWS IN"
680 DISPLAY AT(11,1):"MATRIX Y (";RY;")."
690 STOP
700 RETURN
710 REM ENTER DATA
720 CALL ENTER("X",RX,CX,X(,))
730 CALL ENTER("Y",RY,CY,Y(,))
740 RETURN
750 REM EDIT DATA
760 CALL EDIT("X",RX,CX,X(,))
770 CALL EDIT("Y",RY,CY,Y(,))
780 RETURN
790 REM COMPUTE
800 CALL CLEAR
```

```
810 DISPLAY AT(12,1):"COMPUTING ..."
820 FOR I=1 TO RX
830 FOR J=1 TO CY
840 Z(I,J)=0
850 FOR L=1 TO CX
860 Z(I,J)=Z(I,J)+X(I,L)*Y(L,J)
870 NEXT L
880 NEXT J
890 NEXT I
900 RETURN
910 REM DISPLAY RESULT
920 REM IN 10 BY 2 BLOCKS
930 FOR Q=1 TO RX STEP 10
940 GOSUB 970
950 NEXT Q
960 RETURN
970 REM PRINT
980 IMAGE ########.###
990 FOR I=1 TO CY STEP 2
1000 CALL CLEAR
1010 CALL HCHAR(1,3,61,28)
1020 DISPLAY AT(2,7):"MATRIX PRODUCT:"
1030 CALL HCHAR(3,3,61,28)
1040 REM COLUMN HEADING
1050 ROW=8 :: COL=7
1060 FOR L=I TO I+1
1070 IF L<=CY THEN DISPLAY AT(5,COL):"COLUMN";L
1080 COL=COL+13
1090 NEXT L
1100 REM VALUES
1110 FOR J=Q TO Q+9
1120 IF J>RX THEN 1200
1130 DISPLAY AT(ROW,1):"R"&STR$(J)
1140 COL=4
1150 FOR L=I TO I+1
1160 IF L<=CY THEN DISPLAY AT(ROW,COL):USING 980:Z
     (J,L)
1170 COL=COL+13
1180 NEXT L
1190 ROW=ROW+1
1200 NEXT J
1210 CALL HCHAR(23,3,61,28)
1220 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1230 ACCEPT AT(24,26):Z$
1240 NEXT I
1250 RETURN
```

```
1260 SUB SZE(F$,T$,N,MX)
1270 CALL CLEAR
1280 DISPLAY AT(1,1):"HOW MANY ";F$;" ARE IN"
1290 DISPLAY AT(2,1):"MATRIX ";T$;" = ? "
1300 ACCEPT AT(2,14)BEEP VALIDATE(DIGIT,""):N$
1310 IF N$="" THEN 1300 ELSE N=VAL(N$)
1320 IF N=0 THEN DISPLAY AT(23,1):"PLEASE ENTER 1
     OR MORE" :: GOTO 1300
1330 IF N>MX THEN DISPLAY AT(23,1):"ONLY";MX;F$;"
     ALLOWED" :: GOTO 1300
1340 SUBEND
1350 SUB ENTER(T$,R,C,Z(,))
1360 FOR I=1 TO C
1370 CALL CLEAR
1380 DISPLAY AT(1,1):"PLEASE ENTER DATA FOR"
1390 DISPLAY AT(2,1):"COLUMN #";I;"OF MATRIX ";T$;
     "."
1400 FOR J=1 TO R
1410 DISPLAY AT(4,1):"ROW #";J;TAB(10);"= ? "
1420 ACCEPT AT(4,14)VALIDATE(NUMERIC,""):X$
1430 IF X$="" THEN Z(J,I)=0 ELSE Z(J,I)=VAL(X$)
1440 NEXT J
1450 NEXT I
1460 SUBEND
1470 SUB EDIT(T$,R,C,Z(,))
1480 FOR I=1 TO C
1490 FOR L=0 TO INT((R-1)/10)
1500 REM DISPLAY DATA
1510 CALL CLEAR
1520 DISPLAY AT(1,1):"THESE ARE VALUES OF ";T$
1530 DISPLAY AT(2,1):"FOR COLUMN";I;":"
1540 FOR J=1 TO 10
1550 M=J+L*10
1560 IF M<=R THEN DISPLAY AT(J+3,1):"ROW(";M;TAB(9
     );")= ";Z(M,I)
1570 NEXT J
1580 REM CORRECT DATA
1590 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
1600 ACCEPT AT(16,20)VALIDATE("YN","")SIZE(1)BEEP:
     S$
1610 IF S$="N" THEN 1730
1620 IF S$="" THEN 1600
1630 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
1640 DISPLAY AT(19,1):"ROW TO BE CORRECTED ?"
1650 ACCEPT AT(19,23)BEEP VALIDATE(DIGIT,""):Q$
1660 IF Q$="" THEN 1650 ELSE Q=VAL(Q$)
```

```
1670 IF Q<(1+L*10)OR Q>R OR Q>(10+L*10)THEN DISPLA
     Y AT(21,1):"PLEASE ENTER ROW WITHIN" :: DISPL
     AY AT(22,1):"BOUNDS SHOWN." :: GOTO 1650
1680 DISPLAY AT(21,1):"WHAT SHOULD THE"
1690 DISPLAY AT(22,1):"VALUE BE ?"
1700 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1710 IF S$="" THEN Z(Q,I)=0 ELSE Z(Q,I)=VAL(S$)
1720 GOTO 1510
1730 NEXT L
1740 NEXT I
1750 SUBEND
```

Sweet and Simple Matrix Inversion

In arithmetic, the reciprocal or inverse of a number is another number which, when multiplied by the original, gives the value 1. For example, the inverse of 2 is 1/2, since 2 * 1/2 = 1. In more general terms, the inverse of X is 1/X.

In matrix math, an inverse is defined similarly. The inverse of matrix X is another matrix Y such that X * Y = I, where I is an identity matrix. An identity matrix is an array with 1's along the principal diagonal—the diagonal running from the upper left to the lower right—and 0's everywhere else. An identity matrix is always square, and only a square matrix can be inverted.

As noted in the section on multiplication, the matrix product X*Y will not usually equal the matrix product Y*X, even when both expressions are defined. One exception to this rule is when Y equals the inverse of X. In this case, Y is usually written X^{-1} (read X *inverse*), with $X*X^{-1} = X^{-1}*X = I$, as Figure 5-5 illustrates.

Figure 5-5. Inverse of a Matrix

$$\underbrace{\begin{bmatrix} 2 & 4 \\ 4 & 6 \end{bmatrix}}_{X} * \underbrace{\begin{bmatrix} -\frac{3}{2} & 1 \\ 1 & -\frac{1}{2} \end{bmatrix}}_{\text{Inverse}} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{I}, \text{ and}$$

$$\underbrace{\begin{bmatrix} -\frac{3}{2} & 1 \\ 1 & -\frac{1}{2} \end{bmatrix}}_{\text{Inverse}} * \underbrace{\begin{bmatrix} 2 & 4 \\ 4 & 6 \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{I}$$

The inverse of a matrix can be computed in several different ways. The method used here—one of the simpler ones—tacks an identity matrix onto the array and transforms the left half of this augmented matrix into I, using elementary row operations. What was once the identity matrix becomes the inverse. If this sounds complicated, don't worry. As Figure 5-6 shows, the procedure is fairly straightforward.

Figure 5-6. The Mechanics of Matrix Inversion

Perform the following steps to invert a matrix:

1. Tack an identity matrix onto the array that you want to invert.

$$\underbrace{\begin{bmatrix} 2 & 4 \\ 1 & 4 \end{bmatrix}}_{X} \quad \underbrace{\left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right]}_{I}$$

2. Transform X into I using elementary row operations.
   a. Divide the first row by the pivot element, 2.

$$\left[\begin{array}{cc|cc} 1 & 2 & \frac{1}{2} & 0 \\ 1 & 4 & 0 & 1 \end{array}\right]$$

   b. Subtract the first row from the second.

$$\left[\begin{array}{cc|cc} 1 & 2 & \frac{1}{2} & 0 \\ 0 & 2 & -\frac{1}{2} & 1 \end{array}\right]$$

   c. Divide the second row by the pivot element, 2.

$$\left[\begin{array}{cc|cc} 1 & 2 & \frac{1}{2} & 0 \\ 0 & 1 & -\frac{1}{4} & \frac{1}{2} \end{array}\right]$$

   d. Finally, multiply the second row by 2 and subtract the product from the first row.

$$\left[\begin{array}{cc|cc} 1 & 0 & 1 & -1 \\ 0 & 1 & -\frac{1}{4} & \frac{1}{2} \end{array}\right]$$

$$\underbrace{\qquad\qquad}_{\text{Inverse}}$$

This algorithm is without bells and whistles. It is designed to handle those cases where a quick solution is required. For greater precision, try the full-fledged Gauss-Jordan Sweep with Complete Pivoting routine at the end of this chapter.

Program 5-3. Sweet and Simple Matrix Inversion

```
100 REM INVERTING A MATRIX USING GAUSS-JORDAN SWEE
    P, WITHOUT PIVOTING
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 420
170 REM INVERT MATRIX
180 GOSUB 530
190 REM DISPLAY RESULT
200 GOSUB 830
210 END
220 REM INITIALIZE
230 DATA 25
240 DIM X(25,50)
250 READ MAXK
260 CALL CLEAR :: DISPLAY AT(1,1):"THIS PROGRAM IN
    VERTS A"
270 DISPLAY AT(2,1):"MATRIX USING GAUSS-JORDAN"
280 DISPLAY AT(3,1):"SWEEP, WITHOUT PIVOTING."
290 DISPLAY AT(5,1):"PLEASE ENTER THE ORDER OF"
300 DISPLAY AT(6,1):"YOUR MATRIX, THAT IS, THE"
310 DISPLAY AT(7,1):"NUMBER OF ROWS OR COLUMNS"
320 DISPLAY AT(8,1):"IN IT."
330 DISPLAY AT(10,1):"REMEMBER, HOWEVER, THAT"
340 DISPLAY AT(11,1):"ONLY A SQUARE MATRIX CAN BE"
350 DISPLAY AT(12,1):"INVERTED."
360 DISPLAY AT(14,1):"ORDER = ?"
370 ACCEPT AT(14,11)BEEP VALIDATE(DIGIT,""):K$
380 IF K$="" THEN 370 ELSE K=VAL(K$)
390 IF K=0 THEN DISPLAY AT(24,1):"THAT'S PRETTY TI
    NY !" :: GOTO 370
400 IF K>MAXK THEN DISPLAY AT(23,1):"PLEASE CHANGE
     LINES 230 AND" :: DISPLAY AT( 24,1):"240 FOR A
     VALUE THAT LARGE." :: STOP
410 RETURN
420 REM ENTER DATA
430 FOR I=1 TO K
440 CALL CLEAR
450 DISPLAY AT(1,1):"PLEASE ENTER DATA FOR"
460 DISPLAY AT(2,1):"COLUMN #";I;"."
470 FOR J=1 TO K
480 DISPLAY AT(4,1):"ROW #";J;TAB(10);"= ?"
```

```
490 ACCEPT AT(4,14)VALIDATE(NUMERIC):X(J,I)
500 NEXT J
510 NEXT I
520 RETURN
530 REM INVERT
540 CALL CLEAR
550 DISPLAY AT(12,1):"INVERTING ..."
560 REM TACK ON IDENTITY
570 GOSUB 610
580 REM COMPUTE
590 GOSUB 680
600 RETURN
610 REM TACK ON I
620 FOR I=1 TO K
630 FOR J=1 TO K
640 IF J<>I THEN X(I,K+J)=0 ELSE X(I,K+J)=1
650 NEXT J
660 NEXT I
670 RETURN
680 REM COMPUTE
690 FOR I=1 TO K
700 PIVOT=X(I,I)
710 FOR J=1 TO 2*K
720 X(I,J)=X(I,J)/PIVOT
730 NEXT J
740 REM ADJUST REMAINING ROWS
750 FOR J=1 TO K
760 C=X(J,I)
770 FOR L=I TO 2*K
780 IF J<>I THEN X(J,L)=X(J,L)-C*X(I,L)
790 NEXT L
800 NEXT J
810 NEXT I
820 RETURN
830 REM DISPLAY RESULT
840 REM IN 10 BY 2 BLOCKS
850 FOR Q=1 TO K STEP 10
860 GOSUB 890
870 NEXT Q
880 RETURN
890 REM PRINT
900 IMAGE ########.###
910 FOR I=1 TO K STEP 2
920 CALL CLEAR
930 CALL HCHAR(1,3,61,28)
940 DISPLAY AT(2,7):"MATRIX INVERSE:"
950 CALL HCHAR(3,3,61,28)
960 REM PRINT COLUMN HEADING
970 ROW=8 :: COL=7
```

```
980 FOR L=I TO I+1
990 IF L<=K THEN DISPLAY AT(5,COL):"COLUMN";L
1000 COL=COL+13
1010 NEXT L
1020 REM VALUES
1030 FOR J=Q TO Q+9
1040 IF J>K THEN 1120
1050 DISPLAY AT(ROW,1):"R"&STR$(J)
1060 COL=4
1070 FOR L=I TO I+1
1080 IF L<=K THEN DISPLAY AT(ROW,COL):USING 900:X(
     J,K+L)
1090 COL=COL+13
1100 NEXT L
1110 ROW=ROW+1
1120 NEXT J
1130 CALL HCHAR(23,3,61,28)
1140 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1150 ACCEPT AT(24,26):Z$
1160 NEXT I
1170 RETURN
```

Determinant of a Matrix

Matrices are often used to solve sets of linear and nonlinear equations and to perform regression analysis. In each of these cases, inversion of a matrix is usually required. But what determines if this is possible? It is decided by the *determinant.*

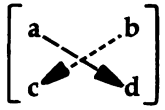
A determinant is a numerical characteristic of a matrix. It is a single value rather than a rectangular grouping of figures. A determinant may be a number like 5 or −22.7 or an expression like 3a + b. Determinants are defined only for square matrices, just as only square matrices can be inverted. If the determinant is 0, the matrix is singular and can't be inverted.

The symbol for the determinant of a matrix is usually two vertical line segments, such as |X| or |Y|. The expression det(X) is also encountered, but less often.

Determinants are computed using well-established rules which vary according to the size of the matrix. For a second-order array, or one with two rows and two columns, it's an easy computation. Figure 5-7 shows how it's done. The determinant in this case equals the product of elements along the principal diagonal minus the ones along the off-diagonal. The principal diagonal slants from upper left to lower right, while the off-diagonal slants the other way.

Figure 5-7. Computation of a Second-Order Determinant

Procedure:



$$= a*d - b*c.$$

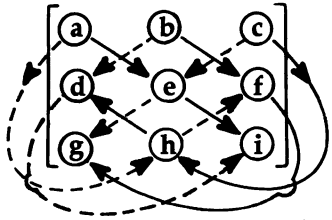
Example:

$$\begin{bmatrix} 2 & 4 \\ 3 & 7 \end{bmatrix} = 2*7 - 4*3 = 2.$$

But calculating a third-order determinant is much more difficult, as Figure 5-8 shows, and for fourth-order or higher matrices it becomes pure penance. Indeed, working out a fourth-order determinant just once is usually one time too many.

Figure 5-8. Computation of a Third-Order Determinant

Procedure:



$$= a*e*i + b*f*g + c*h*d - a*h*f - b*d*i - c*e*g$$

Plus signs are associated with the three products formed by solid lines. Minus signs are associated with the three products formed by dotted lines.

Example:

$$\begin{bmatrix} 5 & 2 & 0 \\ 1 & 3 & 4 \\ 6 & 7 & 8 \end{bmatrix} = 5*3*8 + 2*4*6 + 0*7*1 - 5*7*4 - 2*1*8 - 0*3*6 = 12.$$

Fortunately, your TI computer can use the upper-triangular technique to do the hard work for you. It tallies the determinant of a matrix of any order in two basic steps, as Figure 5-9 shows.

Figure 5-9. The Upper-Triangular Method

Step 1: Transform the elements of the lower triangle into 0's.

$$\begin{bmatrix} 2 & 4 & 8 \\ 1 & 4 & 2 \\ 2 & 4 & 7 \end{bmatrix}$$

To transform the 1 into a 0, add to the second row $-\frac{1}{2}$ times the first row:

$$\begin{bmatrix} 2 & 4 & 8 \\ 0 & 2 & -2 \\ 2 & 4 & 7 \end{bmatrix}$$

To transform the 2 (of the triangle) into a 0, add to the last row -1 times the first row:

$$\begin{bmatrix} 2 & 4 & 8 \\ 0 & 2 & -2 \\ 0 & 0 & -1 \end{bmatrix}$$

Notice that the 4 (of the triangle) coincidentally became a 0 in this last step.

Step 2: Tally the product of principal diagonal elements.

Determinant = 2*2(−1) = −4

Program 5-4. Determinant of a Matrix

```
100 REM DETERMINANT
130 REM INITIALIZE
140 GOSUB 240
150 REM ENTER DATA
160 GOSUB 540
170 REM EDIT DATA
180 GOSUB 660
190 REM COMPUTE DETERMINANT
200 GOSUB 1000
210 REM DISPLAY RESULT
220 GOSUB 1420
230 END
240 REM INITIALIZE
250 REM HEADING
260 GOSUB 300
270 REM ENTER ORDER
280 GOSUB 450
290 RETURN
300 REM HEADING
310 CALL CLEAR
320 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
330 DISPLAY AT(2,1):"DETERMINANT OF A MATRIX."
340 REM MAX SIZE
350 DATA 25
360 DIM X(25,25)
370 READ MSIZE
380 DISPLAY AT(4,1):"THE MAXIMUM ALLOWABLE ORDER"
390 DISPLAY AT(5,1):"OF YOUR MATRIX (NUMBER OF" ::
     DISPLAY AT(6,1):"ROWS AND COLUMNS) IS";MSIZE;
    "."
400 DISPLAY AT(8,1):"CHANGE LINES 350 & 360"
410 DISPLAY AT(9,1):"FOR A DIFFERENT VALUE."
420 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
430 ACCEPT AT(23,25):Z$
440 RETURN
450 REM ORDER
460 CALL CLEAR
470 DISPLAY AT(1,1):"WHAT IS THE ORDER OF YOUR"
480 DISPLAY AT(2,1):"MATRIX ? "
490 ACCEPT AT(2,10)BEEP VALIDATE(NUMERIC,""):K$
500 IF K$="" THEN 490 ELSE K=VAL(K$)
510 IF K<1 THEN DISPLAY AT(23,1):"TOO SMALL." :: D
    ISPLAY AT(24,1):"PLEASE TRY AGAIN." :: GOTO 49
    0
520 IF K>MSIZE THEN DISPLAY AT(23,1):"ONLY ORDER";
    MSIZE;"OR SMALLER IS" :: DISPLAY AT(24,1):"ALL
    OWED. PLEASE TRY AGAIN." :: GOTO 490
530 RETURN
```

```
540 REM ENTER DATA
550 FOR I=1 TO K
560 CALL CLEAR
570 DISPLAY AT(1,1):"PLEASE ENTER DATA FOR"
580 DISPLAY AT(2,1):"COLUMN #";I;"."
590 FOR J=1 TO K
600 DISPLAY AT(4,1):"ROW #";J;TAB(10);"= ? "
610 ACCEPT AT(4,14)VALIDATE(NUMERIC,""):X$
620 IF X$="" THEN X(J,I)=0 ELSE X(J,I)=VAL(X$)
630 NEXT J
640 NEXT I
650 RETURN
660 REM EDIT DATA
670 FOR I=1 TO K
680 FOR L=0 TO INT((K-1)/10)
690 REM DISPLAY DATA
700 GOSUB 760
710 REM CORRECT DATA
720 GOSUB 840
730 NEXT L
740 NEXT I
750 RETURN
760 REM DISPLAY DATA
770 CALL CLEAR
780 DISPLAY AT(1,1):"THESE ARE VALUES OF" :: DISPL
    AY AT(2,1):"COLUMN";I;":"
790 FOR J=1 TO 10
800 M=J+L*10
810 IF M<=K THEN DISPLAY AT(J+3,1):"ROW(";M;TAB(9)
    ;")= ";X(M,I)
820 NEXT J
830 RETURN
840 REM CORRECT DATA
850 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
860 ACCEPT AT(16,20)VALIDATE("YN","")SIZE(1)BEEP:S
    $
870 IF S$="N" THEN 990
880 IF S$="" THEN 860
890 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
900 DISPLAY AT(19,1):"ROW TO BE CORRECTED ?"
910 ACCEPT AT(19,23)BEEP VALIDATE(DIGIT,""):Q$
920 IF Q$="" THEN 910 ELSE Q=VAL(Q$)
930 IF Q<(1+L*10)OR Q>K OR Q>(10+L*10)THEN DISPLAY
     AT(21,1):"PLEASE ENTER ROW WITHIN" :: DISPLAY
     AT(22,1):"BOUNDS SHOWN." :: GOTO 910
940 DISPLAY AT(21,1):"WHAT SHOULD THE"
950 DISPLAY AT(22,1):"VALUE BE ?"
960 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
970 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
980 GOSUB 770 :: GOTO 850
```

```
990 RETURN
1000 REM COMPUTE
1010 CALL CLEAR
1020 DISPLAY AT(12,1):"COMPUTING DETERMINANT ..."
1030 REM 1ST-ORDER DETERMINANT
1040 IF K=1 THEN DET=X(1,1)
1050 REM FILL LOWER TRIANGLE WITH 0's
1060 SIGN=1
1070 IF K<>1 THEN GOSUB 1110
1080 REM TALLY PRODUCT OF PRINCIPAL DIAGONAL ELEME
     NTS
1090 IF K<>1 THEN GOSUB 1350
1100 RETURN
1110 REM FILL LOWER TRIANGLE
1120 FOR L=1 TO K-1
1130 FOR I=L TO K-1
1140 REM SWITCH ROWS TO AVOID DIVISION BY ZERO
1150 IF X(L,L)=0 THEN GOSUB 1240
1160 IF SWT$="NOT SWITCHED" THEN I=K-1 :: L=K-1 ::
      GOTO 1210
1170 C=X(I+1,L)/X(L,L)
1180 FOR J=1 TO K
1190 X(I+1,J)=X(I+1,J)-X(L,J)*C
1200 NEXT J
1210 NEXT I
1220 NEXT L
1230 RETURN
1240 REM SWITCH ROWS
1250 SWT$="NOT SWITCHED"
1260 FOR M=L+1 TO K
1270 IF X(M,L)=0 THEN 1330
1280 FOR Q=1 TO K
1290 HOLD=X(L,Q):: X(L,Q)=X(M,Q):: X(M,Q)=HOLD
1300 NEXT Q
1310 SIGN=-SIGN
1320 SWT$="SWITCHED" :: M=K
1330 NEXT M
1340 RETURN
1350 REM TALLY PRODUCT
1360 PROD=X(1,1)
1370 FOR I=2 TO K
1380 PROD=PROD*X(I,I)
1390 NEXT I
1400 DET=PROD*SIGN
1410 RETURN
1420 REM DISPLAY RESULT
1430 CALL CLEAR
1440 CALL HCHAR(1,3,61,28)
1450 DISPLAY AT(2,11):"RESULTS"
1460 CALL HCHAR(3,3,61,28)
```

```
147Ø DISPLAY AT(5,1):"THE DETERMINANT OF YOUR"
148Ø DISPLAY AT(6,1):"MATRIX IS ";DET
149Ø IF DET<>Ø THEN 153Ø
15ØØ DISPLAY AT(8,1):"THIS MEANS THAT YOUR MATRIX"
151Ø DISPLAY AT(9,1):"IS SINGULAR AND CAN'T BE"
152Ø DISPLAY AT(1Ø,1):"INVERTED."
153Ø CALL HCHAR(23,3,61,28)
154Ø DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
155Ø ACCEPT AT(24,26):Z$
156Ø RETURN
```

Matrix Inversion Using Gauss-Jordan Sweep with Complete Pivoting

This routine is designed to maximize computational accuracy in the matrix inversion process. The additional precision, however, does not come for free. The algorithm is more complex than the "Sweet and Simple" routine presented earlier. "Sweet and Simple" is actually the "Gauss-Jordan Sweep with Complete Pivoting" *without* pivoting. The big difference is that pivoting switches rows and columns to insure that each *pivot element* of the matrix is always of the highest possible absolute value. In this way, round-off error is minimized. A pivot element, incidentally, lies along the principal diagonal and is eventually transformed into the value 1.

The complete pivoting routine takes longer to run than the Sweet and Simple routine. But the additional time is often worth it, especially when the matrix to be inverted is *ill-conditioned*, or nearly singular.

The Gauss-Jordan algorithm is named after Carl Frederich Gauss, a German scholar of the nineteenth century, and for the French mathematician Camille Jordan. Gauss, a mathematical prodigy, frequently amazed his elders with his insight. At the age of ten, for example, he and his classmates were asked to sum the integers from 1 to 100, inclusive, in one of those "I bet that will keep them busy" exercises. Carl Frederich, however, had different ideas. He immediately realized that 1 + 100 = 101, that 2 + 99 = 101, that 3 + 98 = 101, and so on, and that 50 such sums were in the sequence he was supposed to add. He immediately chalked "5050," or 50 x 101, on his slate, proudly proclaiming, "There it lies."

He was the only one in his class to get the problem right.

Program 5-5. Matrix Inversion Using Gauss-Jordan Sweep with Complete Pivoting

```
100 REM MATRIX INVERSION
110 REM GAUSS-JORDAN SWEEP WITH COMPLETE PIVOTING
140 REM INITIALIZE
150 GOSUB 250
160 REM ENTER DATA
170 GOSUB 570
180 REM EDIT DATA
190 GOSUB 690
200 REM INVERT
210 GOSUB 1030
220 REM DISPLAY RESULT
230 GOSUB 2030
240 END
250 REM INITIALIZE
260 REM HEADING
270 GOSUB 310
280 REM ENTER ORDER
290 GOSUB 480
300 RETURN
310 REM HEADING
320 CALL CLEAR
330 DISPLAY AT(1,1):"THIS PROGRAM INVERTS A"
340 DISPLAY AT(2,1):"MATRIX USING GAUSS-JORDAN"
350 DISPLAY AT(3,1):"SWEEP WITH COMPLETE" :: DISPL
    AY AT(4,1):"PIVOTING."
360 REM MAX SIZE
370 DATA 20
380 DIM X(20,40),MEM(20)
390 READ MSIZE
400 DISPLAY AT(6,1):"THE MAXIMUM ALLOWABLE ORDER"
410 DISPLAY AT(7,1):"OF YOUR MATRIX (NUMBER OF"
420 DISPLAY AT(8,1):"ROWS AND COLUMNS) IS";MSIZE;"
    ."
430 DISPLAY AT(10,1):"CHANGE LINES 370 & 380"
440 DISPLAY AT(11,1):"FOR A DIFFERENT VALUE."
450 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
460 ACCEPT AT(23,25):Z$
470 RETURN
480 REM ORDER
490 CALL CLEAR
500 DISPLAY AT(1,1):"WHAT IS THE ORDER OF YOUR"
510 DISPLAY AT(2,1):"MATRIX ? "
520 ACCEPT AT(2,10)BEEP VALIDATE(NUMERIC,""):K$
530 IF K$="" THEN 520 ELSE K=VAL(K$)
540 IF K<1 THEN DISPLAY AT(23,1):"TOO SMALL." :: D
    ISPLAY AT(24,1):"PLEASE TRY AGAIN." :: GOTO 52
    0
```

```
550 IF K>MSIZE THEN DISPLAY AT(23,1):"ONLY ORDER";
    MSIZE;"OR SMALLER IS" :: DISPLAY AT(24,1):"ALL
    OWED. PLEASE TRY AGAIN." :: GOTO 520
560 RETURN
570 REM ENTER DATA
580 FOR I=1 TO K
590 CALL CLEAR
600 DISPLAY AT(1,1):"PLEASE ENTER DATA FOR"
610 DISPLAY AT(2,1):"COLUMN #";I;"."
620 FOR J=1 TO K
630 DISPLAY AT(4,1):"ROW #";J;TAB(10);"= ? "
640 ACCEPT AT(4,14)VALIDATE(NUMERIC,""):X$
650 IF X$="" THEN X(J,I)=0 ELSE X(J,I)=VAL(X$)
660 NEXT J
670 NEXT I
680 RETURN
690 REM EDIT DATA
700 FOR I=1 TO K
710 FOR L=0 TO INT((K-1)/10)
720 REM DISPLAY DATA
730 GOSUB 790
740 REM CORRECT DATA
750 GOSUB 870
760 NEXT L
770 NEXT I
780 RETURN
790 REM DISPLAY DATA
800 CALL CLEAR
810 DISPLAY AT(1,1):"THESE ARE VALUES OF" :: DISPL
    AY AT(2,1):"COLUMN";I;":"
820 FOR J=1 TO 10
830 M=J+L*10
840 IF M<=K THEN DISPLAY AT(J+3,1):"ROW(";M;TAB(9)
    ;")= ";X(M,I)
850 NEXT J
860 RETURN
870 REM CORRECT DATA
880 DISPLAY AT(16,1):"CORRECTIONS (Y/N) ?"
890 ACCEPT AT(16,20)VALIDATE("YN","")SIZE(1)BEEP:S
    $
900 IF S$="N" THEN 1020
910 IF S$="" THEN 890
920 DISPLAY AT(18,1):"WHAT IS THE NUMBER OF THE"
930 DISPLAY AT(19,1):"ROW TO BE CORRECTED ?"
940 ACCEPT AT(19,23)BEEP VALIDATE(DIGIT,""):Q$
950 IF Q$="" THEN 940 ELSE Q=VAL(Q$)
960 IF Q<(1+L*10)OR Q>K OR Q>(10+L*10)THEN DISPLAY
     AT(21,1):"PLEASE ENTER ROW WITHIN" :: DISPLAY
     AT(22,1):"BOUNDS SHOWN." :: GOTO 940
970 DISPLAY AT(21,1):"WHAT SHOULD THE"
```

```
980 DISPLAY AT(22,1):"VALUE BE ?"
990 ACCEPT AT(22,11)BEEP VALIDATE(NUMERIC,""):S$
1000 IF S$="" THEN X(Q,I)=0 ELSE X(Q,I)=VAL(S$)
1010 GOSUB 800 :: GOTO 880
1020 RETURN
1030 REM INVERT
1040 CALL CLEAR
1050 DISPLAY AT(12,1):"INVERTING MATRIX ..."
1060 REM INITIALIZE
1070 GOSUB 1130
1080 REM CONVERT ORIGINAL MATRIX TO IDENTITY
1090 GOSUB 1240
1100 REM INTERCHANGE ROWS WHICH CORRESPOND TO SWIT
     CHED COLUMNS
1110 GOSUB 1820
1120 RETURN
1130 REM INITIALIZE
1140 FOR I=1 TO K
1150 FOR J=1 TO K
1160 IF J=I THEN X(I,K+J)=1 ELSE X(I,K+J)=0
1170 NEXT J
1180 NEXT I
1190 REM ASSIGN INITIAL VALUES TO VECTOR WHICH WIL
     L RECORD COLUMN SWITCHES
1200 FOR I=1 TO K
1210 MEM(I)=I
1220 NEXT I
1230 RETURN
1240 REM CONVERT
1250 FOR Q=1 TO K
1260 REM DETERMINE WHICH POTENTIAL KEY ELEMENT IS
     OF HIGHEST ABSOLUTE VALUE
1270 GOSUB 1400
1280 REM SINGULAR MATRIX
1290 IF HIGH=0 THEN CALL CLEAR :: DISPLAY AT(6,1):
     "YOUR MATRIX IS SINGULAR." :: STOP
1300 REM ROW OR COLUMN SWITCHING IS NOT REQUIRED
1310 IF COL=Q AND ROW=Q THEN 1350
1320 REM SWITCH ROWS OR COLUMNS
1330 IF ABS(X(ROW,Q))>=ABS(X(Q,COL))THEN GOSUB 150
     0 ELSE GOSUB 1580
1340 REM ADJUST KEY ROW
1350 GOSUB 1680
1360 REM ADJUST OTHER ROWS
1370 GOSUB 1740
1380 NEXT Q
1390 RETURN
1400 REM FIND HIGHEST KEY ELEMENT
1410 HIGH=0 :: COL=0 :: ROW=0
1420 FOR I=Q TO K
```

```
1430 REM EXAMINE ROW ELEMENTS
1440 IF ABS(X(I,Q))>HIGH THEN HIGH=ABS(X(I,Q)):: R
     OW=I
1450 REM EXAMINE COLUMN ELEMENTS
1460 IF ABS(X(Q,I))>HIGH THEN HIGH=ABS(X(Q,I)):: C
     OL=I
1470 NEXT I
1480 RETURN
1490 REM SWITCH ROWS
1500 FOR I=1 TO 2*K
1510 HOLD=X(ROW,I)
1520 X(ROW,I)=X(Q,I)
1530 X(Q,I)=HOLD
1540 NEXT I
1550 RETURN
1560 REM SWITCH COLUMNS
1570 REM SWITCH
1580 FOR I=1 TO K
1590 HOLD=X(I,COL)
1600 X(I,COL)=X(I,Q)
1610 X(I,Q)=HOLD
1620 NEXT I
1630 REM RECORD SWITCH
1640 HOLD=MEM(Q)
1650 MEM(Q)=MEM(COL)
1660 MEM(COL)=HOLD
1670 RETURN
1680 REM ADJUST KEY ROW
1690 PIVOT=X(Q,Q)
1700 FOR I=Q TO 2*K
1710 X(Q,I)=X(Q,I)/PIVOT
1720 NEXT I
1730 RETURN
1740 REM ADJUST REMAINING ROWS
1750 FOR I=1 TO K
1760 D=X(I,Q)
1770 FOR J=Q TO 2*K
1780 IF I<>Q THEN X(I,J)=X(I,J)-D*X(Q,J)
1790 NEXT J
1800 NEXT I
1810 RETURN
1820 REM INTERCHANGE ROWS WHICH CORRESPOND TO SWIT
     CHED COLUMNS
1830 FOR I=1 TO K
1840 REM SEARCH VECTOR FOR LOCATION OF VALUE I
1850 LC=0
1860 FOR J=1 TO K
1870 IF MEM(J)=I THEN LC=J
1880 NEXT J
1890 REM SWITCHING IS NOT REQUIRED
```

```
1900 IF LC=I THEN 2010
1910 REM INTERCHANGE CORRESPONDING ROWS
1920 FOR J=1 TO K
1930 HOLD=X(I,K+J)
1940 X(I,K+J)=X(LC,K+J)
1950 X(LC,K+J)=HOLD
1960 NFXT J
1970 REM INTERCHANGE VALUES IN VECTOR WHICH RECORD
     S COLUMN SWITCHES
1980 HOLD=MEM(I)
1990 MEM(I)=MEM(LC)
2000 MEM(LC)=HOLD
2010 NEXT I
2020 RETURN
2030 REM DISPLAY RESULT
2040 REM IN 10 BY 2 BLOCKS
2050 FOR Q=1 TO K STEP 10
2060 GOSUB 2090
2070 NEXT Q
2080 RETURN
2090 REM PRINT
2100 IMAGE ########.###
2110 FOR I=1 TO K STEP 2
2120 CALL CLEAR
2130 CALL HCHAR(1,3,61,28)
2140 DISPLAY AT(2,7):"MATRIX INVERSE:"
2150 CALL HCHAR(3,3,61,28)
2160 REM PRINT COLUMN HEADING
2170 ROW=8 :: COL=7
2180 FOR L=I TO I+1
2190 IF L<=K THEN DISPLAY AT(5,COL):"COLUMN";L
2200 COL=COL+13
2210 NEXT L
2220 REM VALUES
2230 FOR J=Q TO Q+9
2240 IF J>K THEN 2320
2250 DISPLAY AT(ROW,1):"R"&STR$(J)
2260 COL=4
2270 FOR L=I TO I+1
2280 IF L<=K THEN DISPLAY AT(ROW,COL):USING 2100:X
     (J,K+L)
2290 COL=COL+13
2300 NEXT L
2310 ROW=ROW+1
2320 NEXT J
2330 CALL HCHAR(23,3,61,28)
2340 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
2350 ACCEPT AT(24,26):Z$
2360 NEXT I
2370 RETURN
```

Chapter 6

Simple Statistics

Simple Statistics

If you're like most people, you're deluged with an incredible quantity of numerical information—everything from batting averages and miles-per-gallon to stock-market prices and monthly utility bills. Here are several programs to help you transform such data into a more useful form:

- "Mean, Variance, and Standard Deviation" computes average value of a set of observations and determines how much those observations are spread out.
- "Relative and Cumulative Frequencies" computes the percentage of observations that fall into each of several different categories.
- "Frequency Plot" graphically displays the preceding percentages.

Mean, Variance, and Standard Deviation

Washington Redskins fullback John "The Diesel" Riggins pounds through the line for a gain of 4.3 yards. Behind powerful blocking by the Hogs, he carries again on second down for 3.7 more. On third and two he picks up the crucial first down, and over the course of a late game-winning drive he carries the ball 12 times to gain the yardage shown in Table 6-1. Pretty impressive.

But how many yards did Riggins gain per carry? Using the program for computing mean, variance, and standard deviation, your TI can help you find out.

The mean is the average value of a series, or the sum of all values divided by the number of observations. To compute average yards per carry, simply enter the number of yards that Riggins gained in each run—3, 3.7, and so on. The TI sums these values, divides by 12, and displays 3.98, the average.

The TI also reveals that the variance and standard deviation for the observations in Table 6-1 are 4.25 and 2.06 respectively. The variance is a measure of the dispersion of observations about the mean. The higher the variance, the greater the spread. The standard deviation is simply the square root of the variance.

Though the mean tells you the average value of a set of data, the variance may actually be a more useful statistic. In the case of a ball carrier's performance, variance is actually a

measure of consistency. For example, consider Redskins halfback Joe Washington. Average the numbers in Table 6-2 and you'll find that he gains an average of 4.19 yards per carry, very close to Riggins' average of 3.98. But with the help of your trusty TI, you find that Washington's variance is 43.41 while Riggins' variance is only 4.25. That means that Washington's performance is more erratic than that of Riggins. In other words, while Washington may gain little or no yardage on one play and 40 yards on the next, Riggins' gains should be much more consistent.

Table 6-1. Number of Yards Gained by John Riggins

Carry	Yards
1	4.3
2	3.7
3	2.0
4	5.0
5	4.1
6	3.8
7	3.9
8	6.8
9	−1.2
10	6.5
11	4.5
12	4.4

Table 6-2. Number of Yards Gained by Joe Washington

Carry	Yards
1	2.0
2	1.3
3	17.0
4	−3.6
5	−2.0
6	10.2
7	2.4
8	11.8
9	2.3
10	0.5

Program 6-1. Mean, Variance, and Standard Deviation

```
100 REM MEAN,VARIANCE, & STANDARD DEVIATION
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 350
170 REM COMPUTE
180 GOSUB 630
190 REM DISPLAY
200 GOSUB 750
210 END
220 REM INITIALIZE
230 REM MAXIMUM NUMBER OF OBSERVATIONS
240 DATA 300
250 DIM X(300)
260 READ MAXN
270 CALL CLEAR
280 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES THE"
290 DISPLAY AT(2,1):"MEAN, VARIANCE, AND STANDARD"
300 DISPLAY AT(3,1):"DEVIATION OF A GROUP OF"
310 DISPLAY AT(4,1):"OBSERVATIONS."
320 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
330 ACCEPT AT(23,25):Z$
340 RETURN
350 REM ENTER DATA
360 REM NUMBER OF OBSERVATIONS
370 GOSUB 410
380 REM VALUES
390 GOSUB 530
400 RETURN
410 REM NUMBER OF OBSERVATIONS
420 CALL CLEAR
430 DISPLAY AT(1,1):"HOW MANY OBSERVATIONS DO"
440 DISPLAY AT(2,1):"YOU HAVE ?"
450 ACCEPT AT(2,12)BEEP VALIDATE(DIGIT,""):N$
460 IF N$="" THEN 450 ELSE N=VAL(N$)
470 IF N<2 THEN DISPLAY AT(21,1):"AT LEAST 2 VALUE
    S NEEDED" :: GOTO 450
480 IF N<=MAXN THEN 520
490 DISPLAY AT(21,1):"SORRY, ONLY";MAXN;"VALUES"
500 DISPLAY AT(22,1):"ARE ALLOWED. CHANGE LINES"
510 DISPLAY AT(23,1):"240 & 250 AND RUN AGAIN." ::
     STOP
520 RETURN
530 REM VALUES
540 CALL CLEAR
550 DISPLAY AT(1,1):"PLEASE ENTER YOUR DATA."
560 IMAGE Value No. ### =
```

```
570 FOR I=1 TO N
580 DISPLAY AT(6,1):USING 560:I
590 ACCEPT AT(6,17)VALIDATE(NUMERIC,""):X$
600 IF X$="" THEN X(I)=0 ELSE X(I)=VAL(X$)
610 NEXT I
620 RETURN
630 REM COMPUTE
640 REM KEY TERMS
650 S=0 :: SQ=0
660 FOR I=1 TO N
670 S=S+X(I)
680 SQ=SQ+X(I)*X(I)
690 NEXT I
700 REM DESIRED VALUES
710 MEAN=S/N
720 VARIANCE=(SQ-S^2/N)/(N-1)
730 STDEV=SQR(VARIANCE)
740 RETURN
750 REM DISPLAY
760 CALL CLEAR
770 CALL HCHAR(1,3,61,28)
780 DISPLAY AT(2,8):"KEY STATISTICS"
790 CALL HCHAR(3,3,61,28)
800 IMAGE = ########.###
810 DISPLAY AT(6,5):"Mean"
820 DISPLAY AT(6,12):USING 800:MEAN
830 DISPLAY AT(8,1):"Variance"
840 DISPLAY AT(8,12):USING 800:VARIANCE
850 DISPLAY AT(10,1):"Standard" :: DISPLAY AT(11,2
    ):"Deviation"
860 DISPLAY AT(11,12):USING 800:STDEV
870 CALL HCHAR(23,3,61,28)
880 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
890 ACCEPT AT(24,26):Z$
900 RETURN
```

Relative and Cumulative Frequencies

Have you ever wondered where the money in your monthly budget really goes? If so, you'll find the next program particularly useful. It computes relative and cumulative frequencies and tells you what percentage of your budget actually goes to different categories.

The program is easy to use. First, enter the number of *classes,* or categories of information. For example, if you wanted to evaluate nine different types of spending, you would type in "9."

Next, enter the *frequency* (the number of dollars spent) for each class. Using the data in Table 6-3, enter $50 for the first class (Movies), $175 for the second class (Clothes), $40 for the third class (Allowance), and so on.

Table 6-3. Monthly Family Spending

Class	Amount
Movies	$ 50
Clothes	175
Allowance	40
Hairdresser	125
Dog Chow	175
Mortgage Payments	950
Transportation	70
Utilities	110
Food	300
	$1995

Your TI then displays the percentages shown in Table 6-4. Total spending for the month was $1995. Each *relative* frequency tells you the percent of the total that was spent in each category.

Table 6-4. Frequencies

Class	Relative Frequency	Cumulative Frequency
Movies	2.51%	2.51%
Clothes	8.77	11.28
Allowance	2.01	13.28
Hairdresser	6.27	19.55
Dog Chow	8.77	28.32
Mortgage Payments	47.62	75.94
Transportation	3.51	79.45
Utilities	5.51	84.96
Food	15.04	100.00

Now assume that you've set up your list so the first five categories represent luxury items. To find out how much was spent for luxuries, you need to know the *cumulative* frequency for the first five categories. The cumulative frequency for any number of categories can be found by summing the relative frequencies for the categories involved. In this case, the cumulative frequency for the first five categories is 28.32 percent.

Program 6-2. Relative and Cumulative Frequencies

```
100 REM RELATIVE & CUMULATIVE FREQUENCIES
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER DATA
160 GOSUB 400
170 REM COMPUTE
180 GOSUB 650
190 REM DISPLAY RESULTS
200 GOSUB 910
210 END
220 REM INITIALIZE
230 CALL CLEAR
240 REM MAX NUMBER OF CLASSES
250 DATA 200
260 DIM F(200),RF(200),CF(200)
270 READ MAXN
280 DISPLAY AT(1,1):"THIS PROGRAM COMPUTES"
290 DISPLAY AT(2,1):"RELATIVE AND CUMULATIVE"
300 DISPLAY AT(3,1):"FREQUENCIES."
310 DISPLAY AT(5,1):"A FREQUENCY IS THE NUMBER OF"
320 DISPLAY AT(6,1):"TIMES THAT AN OBSERVATION"
330 DISPLAY AT(7,1):"OCCURS."
340 DISPLAY AT(9,1):"AND A RELATIVE FREQUENCY IS"
350 DISPLAY AT(10,1):"PERCENTAGE OF TIMES THAT IT"
360 DISPLAY AT(11,1):"OCCURS."
370 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
380 ACCEPT AT(23,25):Z$
390 RETURN
400 REM ENTER DATA
410 CALL CLEAR
420 DISPLAY AT(1,1):"HOW MANY CLASSES DO YOU"
430 DISPLAY AT(2,1):"HAVE ?"
440 ACCEPT AT(2,8)BEEP VALIDATE(DIGIT,""):C$
450 IF C$="" THEN 440 ELSE N=VAL(C$)
460 IF N=0 THEN 440
470 IF N<=MAXN THEN 510
```

```
480 DISPLAY AT(22,1):"SORRY, ONLY";MAXN;"CLASSES"
490 DISPLAY AT(23,1):"ARE ALLOWED. CHANGE LINES"
500 DISPLAY AT(24,1):"250 & 260 AND RUN AGAIN." ::
    STOP
510 REM FREQUENCIES
520 GOSUB 540
530 RETURN
540 REM FREQUENCIES
550 CALL CLEAR
560 DISPLAY AT(1,1):"PLEASE ENTER THE FREQUENCIES"
570 DISPLAY AT(2,1):"FOR EACH CLASS."
580 IMAGE Class No. ### =
590 FOR I=1 TO N
600 DISPLAY AT(6,1):USING 580:I
610 ACCEPT AT(6,17)VALIDATE(NUMERIC,""):V$
620 IF V$="" THEN F(I)=0 ELSE F(I)=VAL(V$)
630 NEXT I
640 RETURN
650 REM COMPUTE
660 CALL CLEAR
670 DISPLAY AT(12,1):"COMPUTING ..."
680 REM RELATIVE FREQUENCIES
690 GOSUB 730
700 REM CUMULATIVE FREQUENCIES
710 GOSUB 840
720 RETURN
730 REM RELATIVE FREQUENCIES
740 REM SUM
750 SUM=0
760 FOR I=1 TO N
770 SUM=SUM+F(I)
780 NEXT I
790 IF SUM=0 THEN DISPLAY AT(14,1):"SORRY, THE SUM
    OF YOUR" :: DISPLAY AT(15,1):"FREQUENCIES EQU
    ALS 0." :: STOP
800 FOR I=1 TO N
810 RF(I)=F(I)*100/SUM
820 NEXT I
830 RETURN
840 REM CUMULATIVE FREQUENCIES
850 CUM=0
860 FOR I=1 TO N
870 CUM=CUM+F(I)
880 CF(I)=CUM*100/SUM
890 NEXT I
900 RETURN
910 REM DISPLAY RESULTS
920 FOR L=0 TO INT((N-1)/10)
930 REM HEADING
```

```
940 GOSUB 990
950 REM BODY
960 GOSUB 1060
970 NEXT L
980 RETURN
990 REM HEADING
1000 CALL CLEAR
1010 CALL HCHAR(1,3,61,28)
1020 DISPLAY AT(2,13):"FREQUENCIES"
1030 DISPLAY AT(3,1):"Class";TAB(9);"Actual";TAB(1
     8);"Rel.";TAB(25);"Cum."
1040 CALL HCHAR(4,3,61,28)
1050 RETURN
1060 REM BODY
1070 IMAGE ### ######.### ###.## ###.##
1080 FOR J=1 TO 10
1090 Q=J+L*10
1100 IF Q<=N THEN DISPLAY AT(J+7,1):USING 1070:Q,F
     (Q),RF(Q),CF(Q)
1110 NEXT J
1120 CALL HCHAR(23,3,61,28)
1130 DISPLAY AT(24,3):"Press any key to continue"
1140 CALL KEY(0,K,S)
1150 IF S=0 THEN 1140
1160 RETURN
```

Frequency Plot

Graphs let you see at a glance what might otherwise be lost in a long column of figures. Indeed, if a picture is worth a thousand words, a good graph may be worth a million.

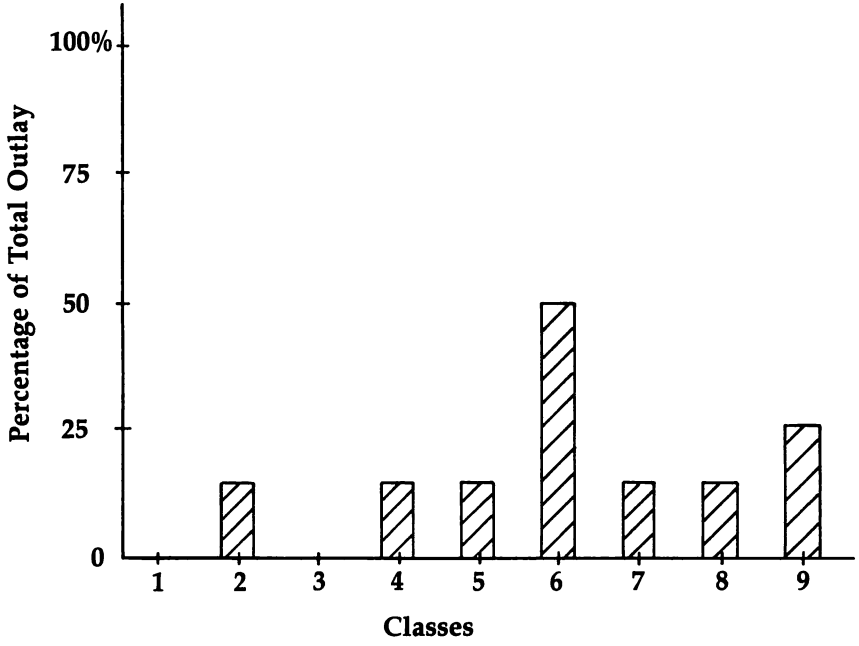
This program plots a relative-frequency distribution on your TI video screen. Input your data just as you did when computing relative and cumulative frequencies. First enter the number of classes or categories of information. Then enter the observations in each.

Using the preceding example, the TI plots the family's expenditure breakdown as shown in Figure 6-1. You immediately notice that Class 6, mortgage payments, towers over all other expenses. Further, Class 1 and Class 3, movies and allowances, are proportionately so small that they don't even show up.

The frequency plot lets you quickly and easily visualize how the family spends its money. Further, important differences in expenditures are emphasized by different-sized

columns, with the height of each drawn according to its percentage of total outlays. This beats looking at a dull column of figures.

Figure 6-1. Relative Frequencies



Program 6-3. Frequency Plot

```
100 REM RELATIVE FREQUENCY PLOT
130 REM INITIALIZE
140 GOSUB 200
150 REM ENTER DATA
160 GOSUB 510
170 REM PLOT
180 GOSUB 770
190 END
200 REM INITIALIZE
210 REM HEADING
220 GOSUB 260
230 REM SPECIAL CHARACTERS
240 GOSUB 390
250 RETURN
260 REM HEADING
```

```
270 CALL CLEAR
280 DISPLAY AT(1,1):"THIS PROGRAM PLOTS RELATIVE"
290 DISPLAY AT(2,1):"FREQUENCIES."
300 DISPLAY AT(4,1):"A FREQUENCY IS THE NUMBER OF"
310 DISPLAY AT(5,1):"TIMES THAT AN OBSERVATION"
320 DISPLAY AT(6,1):"OCCURS."
330 DISPLAY AT(8,1):"AND A RELATIVE FREQUENCY IS"
340 DISPLAY AT(9,1):"THE PERCENTAGE OF TIMES THAT"
350 DISPLAY AT(10,1):"IT OCCURS."
360 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
370 ACCEPT AT(23,25):Z$
380 RETURN
390 REM SPECIAL CHARACTERS
400 REM COLUMNS
410 CALL CHAR(127,"FFFFFFFFFFFFFFFF")
420 CALL CHAR(128,"F0F0F0F0F0F0F0F0")
430 REM Y-AXIS
440 CALL CHAR(129,"E0E0E0E0E0E0E0E0")
450 REM X-AXIS
460 CALL CHAR(130,"FFFFFF0000000000")
470 REM TIC MARKS
480 REM Y-AXIS
490 CALL CHAR(131,"FCE0E0E0E0E0E0E0")
500 RETURN
510 REM ENTER DATA
520 REM CLASSES
530 GOSUB 570
540 REM OBSERVATIONS
550 GOSUB 660
560 RETURN
570 REM CLASSES
580 CALL CLEAR
590 DISPLAY AT(1,1):"HOW MANY CLASSES DO YOU"
600 DISPLAY AT(2,1):"HAVE ?"
610 ACCEPT AT(2,8)BEEP VALIDATE(DIGIT,""):K$
620 IF K$="" THEN 610 ELSE K=VAL(K$)
630 IF K=0 THEN 610
640 IF K>10 THEN DISPLAY AT(23,1):"SORRY, ONLY 10
    ARE ALLOWED" :: GOTO 610
650 RETURN
660 REM OBSERVATIONS
670 CALL CLEAR
680 DISPLAY AT(1,1):"PLEASE ENTER THE FREQUENCIES"
690 DISPLAY AT(2,1):"FOR EACH CLASS."
700 IMAGE Class No. ## =
710 FOR I=1 TO K
720 DISPLAY AT(6,1):USING 700:I
730 ACCEPT AT(6,16)VALIDATE(NUMERIC,""):C$
740 IF C$="" THEN 730 ELSE CLASS(I)=VAL(C$)
```

```
750 NEXT I
760 RETURN
770 REM PLOT
780 REM AXES
790 GOSUB 850
800 REM COLUMNS
810 GOSUB 1160
820 REM HEADING
830 GOSUB 1450
840 RETURN
850 REM AXES
860 CALL CLEAR
870 REM Y
880 CALL VCHAR(3,6,129,16)
890 DISPLAY AT(2,2):"%"
900 REM X
910 CALL HCHAR(19,6,130,23)
920 DISPLAY AT(22,12):"Classes"
930 REM LABEL
940 REM Y-AXIS
950 ROW=19
960 IMAGE ###
970 FOR I=0 TO 100 STEP 25
980 DISPLAY AT(ROW,1)SIZE(3):USING 960:I
990 ROW=ROW-4
1000 NEXT I
1010 ROW=15
1020 FOR I=25 TO 100 STEP 25
1030 CALL VCHAR(ROW,6,131)
1040 ROW=ROW-4
1050 NEXT I
1060 REM X-AXIS
1070 REM SPACES BETWEEN TIC MARKS
1080 S=INT(22/K)
1090 C=5
1100 IMAGE ###
1110 FOR I=1 TO K
1120 DISPLAY AT(20,C):I
1130 C=C+S
1140 NEXT I
1150 RETURN
1160 REM COLUMNS
1170 REM HEIGHT OF EACH BAR
1180 GOSUB 1220
1190 REM PLOT
1200 GOSUB 1360
1210 RETURN
1220 REM HEIGHT
1230 REM PERCENTAGES
```

```
1240 SUM=0
1250 FOR I=1 TO K
1260 SUM=SUM+CLASS(I)
1270 NEXT I
1280 FOR I=1 TO K
1290 PER(I)=CLASS(I)*100/SUM
1300 NEXT I
1310 REM HEIGHT
1320 FOR I=1 TO K
1330 RW(I)=(19-.16*PER(I))
1340 NEXT I
1350 RETURN
1360 REM PLOT
1370 C=8
1380 FOR I=1 TO K
1390 R=RW(I)
1400 CALL VCHAR(R,C,127,19-R)
1410 CALL VCHAR(R,C+1,128,19-R)
1420 C=C+S
1430 NEXT I
1440 RETURN
1450 REM HEADING
1460 DISPLAY AT(1,6):"RELATIVE FREQUENCIES"
1470 DISPLAY AT(24,3):"Press any key to continue"
1480 CALL KEY(0,K,S)
1490 IF S=0 THEN 1480
1500 RETURN
```

Chapter 7

Numerical Analysis

Numerical Analysis

Your TI computer is designed to manipulate large groups of numbers quickly and easily, and this chapter presents several algorithms designed to take advantage of its tremendous number-crunching capabilities. Use these numerical-analysis techniques as they stand or as subroutines in larger programs.

- "Sort" lets you arrange a group of numbers into high-to-low or low-to-high sequence.
- "Random Number Generator" produces numbers with a normal (random) distribution.
- "Random Number Tester" measures the quality of a computer's random number generator.
- "Numerical Integration" computes the area under a curve.
- "Derivative" computes the rate of change of a variable.

Sort

Suppose a teacher wants to sort the test scores given in Table 7-1 into descending sequence—that is, she wants the highest grade first and the lowest grade last. In precomputer days she'd have had to do the job manually. She'd carefully search the list for the highest grade, cross it off, and put it at the top of a separate sheet of paper. Then she'd repeat the procedure for the second highest grade, the third highest grade, and so on, until either the list or her patience was exhausted.

Table 7-1. Test Scores in Natural Order

Score	Score
37	82
52	91
28	83
96	52
86	97
95	93
64	70
75	28
74	61
92	80
98	66
65	69
63	89

With the TI there's an easier way. Simply input your figures into the computer and indicate how you want them sorted—high to low or low to high. Table 7-2 shows the test scores in high-to-low sequence, as desired.

Table 7-2. Test Scores in Descending Order

Score	Score
98	74
97	70
96	69
95	66
93	65
92	64
91	63
89	61
86	52
83	52
82	37
80	28
75	28

Program 7-1. Sort

```
100 REM SORT
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER NUMBERS
160 GOSUB 510
170 REM SORT
180 GOSUB 790
190 REM DISPLAY
200 GOSUB 1050
210 END
220 REM INITIALIZE
230 REM TITLE
240 GOSUB 300
250 REM HEADING
260 GOSUB 390
270 REM ENTER ORDER
280 GOSUB 430
290 RETURN
300 REM TITLE
```

```
310 CALL CLEAR
320 DATA ORTS,RTSO,TSOR,SORT
330 FOR I=1 TO 4
340 READ W$
350 DISPLAY AT(12,12):W$
360 FOR DELAY=1 TO 100 :: NEXT DELAY
370 NEXT I
380 RETURN
390 REM HEADING
400 CALL CLEAR
410 DISPLAY AT(1,1):"THIS PROGRAM SORTS NUMBERS."
420 RETURN
430 REM ORDER
440 DISPLAY AT(4,1):"WOULD YOU LIKE THEM IN:"
450 DISPLAY AT(6,3):"A. LOW TO HIGH, OR"
460 DISPLAY AT(7,3):"B. HIGH TO LOW"
470 DISPLAY AT(10,1):"SEQUENCE (A/B) ?"
480 ACCEPT AT(10,18)BEEP VALIDATE("AB","")SIZE(1):
    C$
490 IF C$="" THEN 480
500 RETURN
510 REM ENTER NUMBERS
520 REM TOTAL NUMBER
530 GOSUB 570
540 REM VALUES
550 GOSUB 700
560 RETURN
570 REM TOTAL NUMBER
580 CALL CLEAR
590 DATA 300
600 DIM X(300)
610 READ MAXN
620 DISPLAY AT(1,1):"HOW MANY VALUES DO YOU WANT"
630 DISPLAY AT(2,1):"TO SORT ?"
640 ACCEPT AT(2,11)BEEP VALIDATE(DIGIT,""):N$
650 IF N$="" THEN 640 ELSE N=VAL(N$)
660 IF N=0 THEN 640
670 IF N>MAXN THEN DISPLAY AT(21,1):"ONLY";MAXN;"V
    ALUES ARE ALLOWED."
680 IF N>MAXN THEN DISPLAY AT(23,1):"CHANGE LINES
    600 & 610 FOR A" :: DISPLAY AT(24,1):"HIGHER L
    IMIT." :: GOTO 640
690 RETURN
700 REM VALUES
710 CALL CLEAR
720 DISPLAY AT(1,1):"PLEASE ENTER YOUR DATA."
730 FOR I=1 TO N
740 DISPLAY AT(5,1):"No.(";I;TAB(10);") = ?"
750 ACCEPT AT(5,16)VALIDATE(NUMERIC,""):S$
760 IF S$="" THEN X(I)=0 ELSE X(I)=VAL(S$)
```

```
770 NEXT I
780 RETURN
790 REM SORT
800 CALL CLEAR
810 DISPLAY AT(12,1):"SORTING ..."
820 REM ORDER
830 IF C$="A" THEN GOSUB 850 ELSE GOSUB 950
840 RETURN
850 REM LOW TO HIGH
860 FOR I=1 TO N
870 FOR J=1 TO N-1
880 IF X(J+1)>=X(J)THEN 920
890 HOLD=X(J)
900 X(J)=X(J+1)
910 X(J+1)=HOLD
920 NEXT J
930 NEXT I
940 RETURN
950 REM HIGH TO LOW
960 FOR I=1 TO N
970 FOR J=1 TO N-1
980 IF X(J+1)<=X(J)THEN 1020
990 HOLD=X(J)
1000 X(J)=X(J+1)
1010 X(J+1)=HOLD
1020 NEXT J
1030 NEXT I
1040 RETURN
1050 REM DISPLAY
1060 FOR L=0 TO INT((N-1)/10)
1070 GOSUB 1100
1080 NEXT L
1090 RETURN
1100 REM 10 VALUES AT A TIME
1110 CALL CLEAR
1120 CALL HCHAR(1,3,61,28)
1130 DISPLAY AT(2,9):"SORTED VALUES"
1140 IF C$="A" THEN S$="Low to High" ELSE S$="High
      to Low"
1150 DISPLAY AT(3,6):"ORDER: ";S$
1160 CALL HCHAR(4,3,61,28)
1170 FOR J=1 TO 10
1180 Q=J+L*10
1190 IF Q<=N THEN DISPLAY AT(J+5,1):"No.(";Q;TAB(1
     0);")= ";X(Q)
1200 NEXT J
1210 CALL HCHAR(23,3,61,28)
1220 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
1230 ACCEPT AT(24,26):Z$
1240 RETURN
```

Random Number Generator

Have you ever seen a pellet-dropping machine? If not, take a look at Figure 7-1. A small pellet is dropped through the hole. It hits the top peg and bounces left or right with equal chance, eventually finding its way into one of the cups below. The spread of pellets among the bottom bins bears close resemblance to what statisticians call *normal distribution,* especially when large numbers of pegs and cups are used.

The normal distribution is the most popular and powerful in statistics. A normal distribution curve is bell-shaped and extends from minus to plus infinity, with mean, mode, and median all equal. The mean is the average value of the distribution and can be thought of as the point where the distribution curve would balance if it could be placed on the head of a pin. The mode is the value of the curve that occurs most often. The median is the middle value: Half of all values lie to the left of the median, and half to the right. Many physical and social phenomena closely approximate normal distribution, including IQ, the weights of adults, and the quantity of soda actually contained in 12-ounce cans. The general form of the distribution curve is shown in Figure 7-2; specific curve shapes depend on the variance of the data, as shown in Figure 7-3.

Figure 7-1. Pellet-Dropping Machine

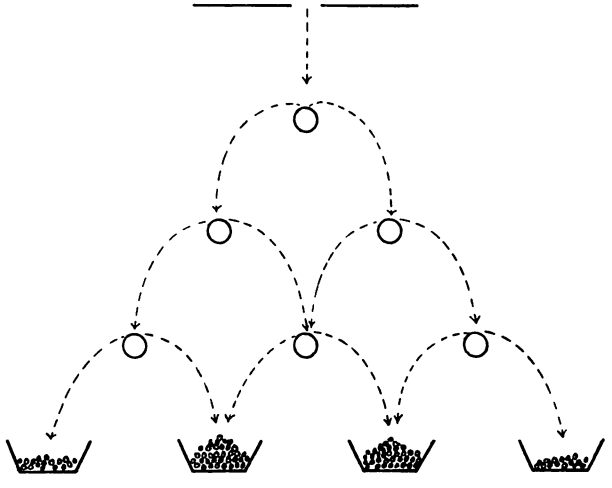


Figure 7-2. The Normal Curve

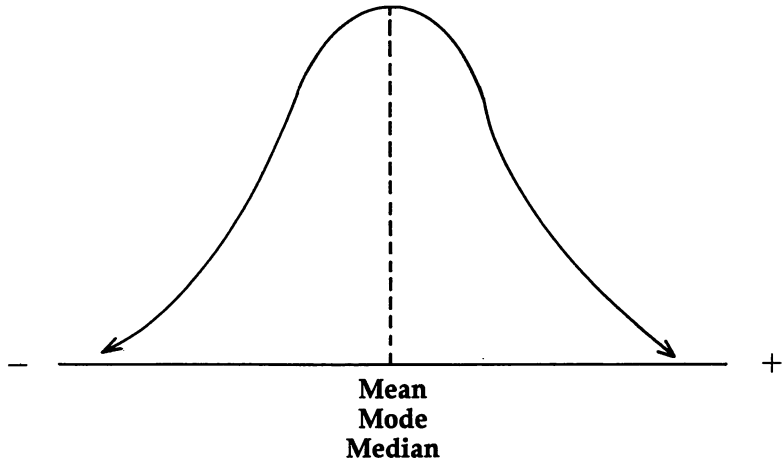
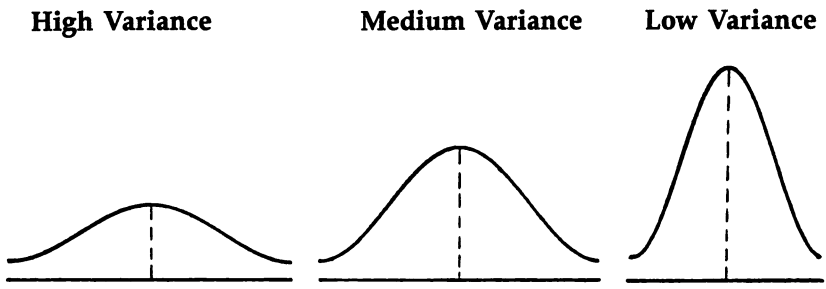


Figure 7-3. Members of the Family of Normal Curves



Random Number Generator produces a list of random numbers from a normal distribution. Start by telling the computer how many values you want to generate, then specify the desired distribution curve by entering the mean and variance. The mean represents the average point of the curve, while the variance describes the distribution of the numbers. High variance values yield number lists that are relatively spread out, while low variance values will tend to cluster the numbers about the mean.

Table 7-3. Random Numbers Generated by the TI from a Normal Distribution with Mean 0 and Variance 10

Number	Number
1.154	0.911
−1.206	7.302
−2.151	−1.017
−5.871	1.523
−3.890	−0.732

Once you've entered the desired parameters, the TI does the rest. Table 7-3, for example, shows a random number list generated from a normal distribution curve with a mean of 0 and a variance of 10.

You *could* do the same thing with the old pellet-dropping machine. But the TI is a lot faster!

Program 7-2. Random Number Generator

```
100 REM RANDOM NUMBER GENERATOR
130 REM INITIALIZE
140 GOSUB 220
150 REM ENTER MEAN & VARIANCE
160 GOSUB 390
170 REM GENERATE VALUES
180 GOSUB 520
190 REM DISPLAY RESULTS
200 GOSUB 690
210 END
220 REM INITIALIZE
230 CALL CLEAR
240 DISPLAY AT(1,1):"THIS PROGRAM GENERATES"
250 DISPLAY AT(2,1):"RANDOM NUMBERS FROM A NORMAL"
260 DISPLAY AT(3,1):"DISTRIBUTION."
270 REM MAXIMUM NUMBER
280 DATA 300
290 DIM X(301)
300 READ MAXN
310 DISPLAY AT(6,1):"HOW MANY VALUES WOULD YOU"
320 DISPLAY AT(7,1):"LIKE TO GENERATE ?"
330 ACCEPT AT(7,20)BEEP VALIDATE(DIGIT,""):N$
340 IF N$="" THEN 330 ELSE N=VAL(N$)
350 IF N=0 THEN 330
```

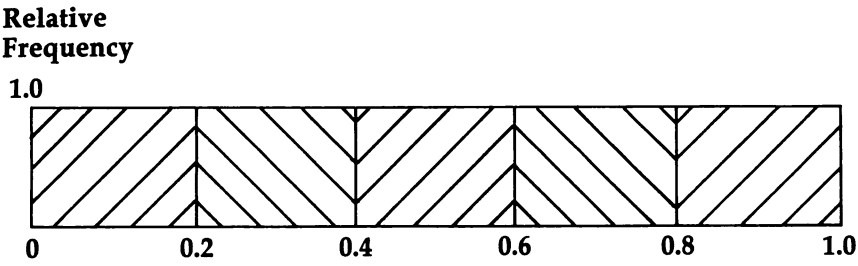
```
360 IF N>MAXN THEN DISPLAY AT(21,1):"ONLY";MAXN;"V
    ALUES ALLOWED."
370 IF N>MAXN THEN DISPLAY AT(23,1):"CHANGE LINES
    280 & 290 FOR" :: DISPLAY AT(24,1):"A HIGHER L
    IMIT." :: GOTO 330
380 RETURN
390 REM MEAN & VARIANCE
400 CALL CLEAR
410 DISPLAY AT(1,1):"PLEASE ENTER THE MEAN AND"
420 DISPLAY AT(2,1):"VARIANCE OF YOUR NORMAL"
430 DISPLAY AT(3,1):"DISTRIBUTION."
440 DISPLAY AT(5,7):"MEAN = ?"
450 ACCEPT AT(5,16)BEEP VALIDATE(NUMERIC,""):M$
460 IF M$="" THEN 450 ELSE M=VAL(M$)
470 DISPLAY AT(6,3):"VARIANCE = ?"
480 ACCEPT AT(6,16)BEEP VALIDATE(NUMERIC,""):V$
490 IF V$="" THEN 480 ELSE V=VAL(V$)
500 IF V<0 THEN DISPLAY AT(23,1):"VARIANCE CAN'T B
    E NEGATIVE." :: GOTO 480
510 RETURN
520 REM GENERATE VALUES
530 CALL CLEAR
540 DISPLAY AT(12,1):"GENERATING VALUES ..."
550 FOR I=1 TO N STEP 2
560 REM SELECT 2 NUMBERS FROM UNIFORM DISTRIBUTION
570 A=RND
580 B=RND
590 REM MAKE INTERMEDIATE CALCULATIONS
600 C=2*A-1
610 D=2*B-1
620 SUM=C*C+D*D
630 IF SUM>=1 THEN 570
640 REM DESIRED NUMBERS
650 X(I)=SQR(V)*C*SQR(-2*LOG(SUM)/SUM)+M
660 X(I+1)=SQR(V)*D*SQR(-2*LOG(SUM)/SUM)+M
670 NEXT I
680 RETURN
690 REM DISPLAY
700 FOR L=0 TO INT((N-1)/10)
710 GOSUB 740
720 NEXT L
730 RETURN
740 REM DISPLAY 10 VALUES AT A TIME
750 CALL CLEAR
760 IMAGE ######.###
770 CALL HCHAR(1,3,61,28)
780 DISPLAY AT(2,8):"RANDOM NUMBERS"
790 CALL HCHAR(3,3,61,28)
800 DISPLAY AT(5,6):"Mean =" :: DISPLAY AT(5,13):U
    SING 760:M
```

```
810 DISPLAY AT(6,2):"Variance =" :: DISPLAY AT(6,1
    3):USING 760:V
820 FOR J=1 TO 10
830 Q=J+L*10
840 IF Q<=N THEN DISPLAY AT(J+8,1):"No.(";Q;TAB(10
    );")=" :: DISPLAY AT(J+8,13):USING 760:X(Q)
850 NEXT J
860 CALL HCHAR(23,3,61,28)
870 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
880 ACCEPT AT(24,26):Z$
890 RETURN
```

Random Number Tester

The TI command RND generates a positive fraction, such as 0.817, 0.513, or 0.926, from a uniform distribution. In this distribution, the chances of drawing a number between 0.0 and 0.1, or between 0.1 and 0.2, or between 0.2 and 0.3, and so on, are all equal. As Figure 7-4 shows, the probability that a given fraction will fall into any one interval is equal to 0.1, or the span of the distribution divided by the number of intervals within it (1/10).

Figure 7-4. Uniform Distribution Between 0 and 1



The TI algorithm which produces those fractions is called a random number generator. The phrase *random number,* however, is actually a misnomer. Strictly speaking, random should modify sequence rather than number. It is the *placement* of fractions within the sequence that is supposed to be random, rather than the numbers themselves.

This computer program tests the quality of the TI's random number generator. With slight modification, it can be used to examine the properties of random number generators on other machines as well.

To use the program, first enter the number of fractions that you want to generate—for instance, 10,000. The TI will then produce 10,000 values and count how many fall into each of 10 cells in the interval 0 to 1. The details concerning one such run are shown in Table 7-4.

Table 7-4. Test Results (10,000 Fractions Generated)

Cell Number	Count	Percent
1	1017	10.17%
2	973	9.73
3	958	9.58
4	1013	10.13
5	1008	10.08
6	971	9.71
7	974	9.74
8	1047	10.47
9	1028	10.28
10	1011	10.11

In theory, each cell should contain 1000 fractions. However, the observed count is almost always higher or lower by some small margin. How small must that margin be for the numbers to be considered random? You use the chi-square (pronounced ki-square) test statistic to find out. This statistic measures how close the actual cell counts are to theoretical values. The better the match, the lower the chi-square value will be.

In this case, the chi-square statistic is 7.65. From published tables, you learn that the critical value for chi-square distribution with 9 degrees of freedom (number of cells minus one) is 14.68. Since the observed statistic is less than this critical value, you can conclude that the sequence of 10,000 fractions is, in fact, random.

Program 7-3. Random Number Tester

```
100 REM RANDOM NUMBER TESTER
130 REM INITIALIZE
140 GOSUB 200
150 REM GENERATE FRACTIONS
160 GOSUB 600
170 REM DISPLAY RESULTS
180 GOSUB 740
190 END
200 REM INITIALIZE
210 REM TITLE
220 GOSUB 280
230 REM HEADING
240 GOSUB 350
250 REM NUMBER OF FRACTIONS TO GENERATE
260 GOSUB 510
270 RETURN
280 REM TITLE
290 CALL CLEAR
300 DISPLAY AT(12,10):"Random"
310 DISPLAY AT(13,11):"number"
320 DISPLAY AT(14,12):"tester"
330 FOR DELAY=1 TO 300 :: NEXT DELAY
340 RETURN
350 REM HEADING
360 CALL CLEAR
370 DISPLAY AT(1,1):"THIS PROGRAM TESTS THE"
380 DISPLAY AT(2,1):"QUALITY OF THE RANDOM NUMBER"
390 DISPLAY AT(3,1):"GENERATOR OF YOUR COMPUTER."
400 DISPLAY AT(6,1):"THE CHI-SQUARE TEST IS"
410 DISPLAY AT(7,1):"PERFORMED:"
420 DISPLAY AT(10,1):"1. RANDOM FRACTIONS ARE"
430 DISPLAY AT(11,1):"{3 SPACES}GROUPED INTO 10"
440 DISPLAY AT(12,1):"{3 SPACES}CATEGORIES."
450 DISPLAY AT(14,1):"2. ACTUAL GROUP COUNTS ARE"
460 DISPLAY AT(15,1):"{3 SPACES}COMPARED TO THEORE
    TICAL"
470 DISPLAY AT(16,1):"{3 SPACES}VALUES."
480 DISPLAY AT(23,1):"HIT 'ENTER' TO CONTINUE"
490 ACCEPT AT(23,25):Z$
500 RETURN
510 REM NUMBER OF FRACTIONS
520 CALL CLEAR
530 DISPLAY AT(1,1):"HOW MANY FRACTIONS DO YOU"
540 DISPLAY AT(2,1):"WANT YOUR COMPUTER TO"
550 DISPLAY AT(3,1):"GENERATE ?"
560 ACCEPT AT(3,12)BEEP VALIDATE(DIGIT,""):N$
570 IF N$="" THEN 560 ELSE N=VAL(N$)
```

```
580 IF N=0 THEN 560
590 RETURN
600 REM GENERATE FRACTIONS
610 DIM CELL(10)
620 FOR I=1 TO 10 :: CELL(I)=0 :: NEXT I
630 CALL CLEAR
640 DISPLAY AT(5,1):"Total Number of Fractions"
650 DISPLAY AT(6,1):"to Generate ="
660 DISPLAY AT(6,14):N
670 DISPLAY AT(12,1):"Count ="
680 FOR I=1 TO N
690 DISPLAY AT(12,8):I
700 K=INT(RND*10)+1
710 CELL(K)=CELL(K)+1
720 NEXT I
730 RETURN
740 REM RESULTS
750 REM CELL COUNT
760 GOSUB 800
770 REM CHI-SQUARE VALUE
780 GOSUB 940
790 RETURN
800 REM CELL COUNT
810 CALL CLEAR
820 CALL HCHAR(1,3,61,28)
830 DISPLAY AT(2,10):"CELL COUNT"
840 CALL HCHAR(3,3,61,28)
850 DISPLAY AT(5,4):"Cell";TAB(12);"Number";TAB(21
    );"Percent"
860 IMAGE No. ##{3 SPACES}#######  ###.## %
870 FOR I=1 TO 10
880 DISPLAY AT(I+7,2):USING 860:I,CELL(I),CELL(I)/
    N*100
890 NEXT I
900 CALL HCHAR(23,3,61,28)
910 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
920 ACCEPT AT(24,26):Z$
930 RETURN
940 REM CHI-SQUARE
950 CHI=0
960 FOR I=1 TO 10
970 CHI=CHI+(CELL(I)-N/10)^2/(N/10)
980 NEXT I
990 CALL CLEAR
1000 CALL HCHAR(5,3,61,28)
1010 DISPLAY AT(6,9):"TEST RESULT"
1020 CALL HCHAR(7,3,61,28)
1030 DISPLAY AT(10,1):"Chi-Square"
1040 DISPLAY AT(11,1):"Statistic ="
```

```
105Ø IMAGE #####.##
106Ø DISPLAY AT(11,12):USING 105Ø:CHI
107Ø CALL HCHAR(23,3,61,28)
108Ø RETURN
```

Numerical Integration

When you graph a variable's change over time—for instance, calories per day or rainfall per month—the area under the resulting curve is the *total* value of that variable. That is, the area represents total calories, or total rainfall, or total production. The process of computing this area is called integration, and the area is called the integral.

Figure 7-5. Sales Per Day

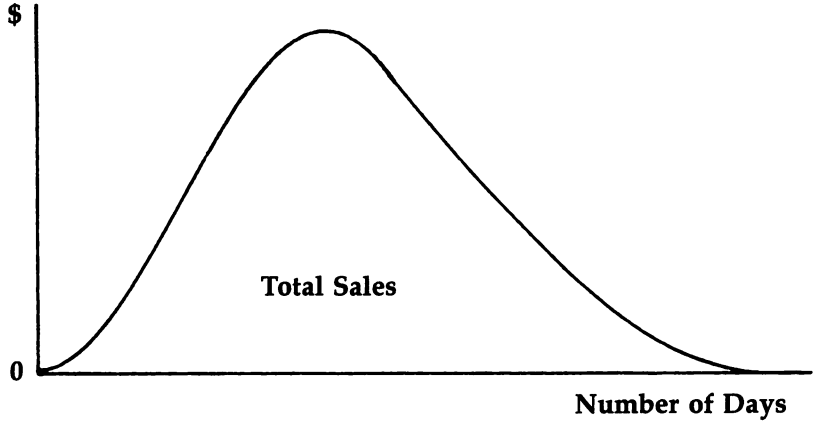
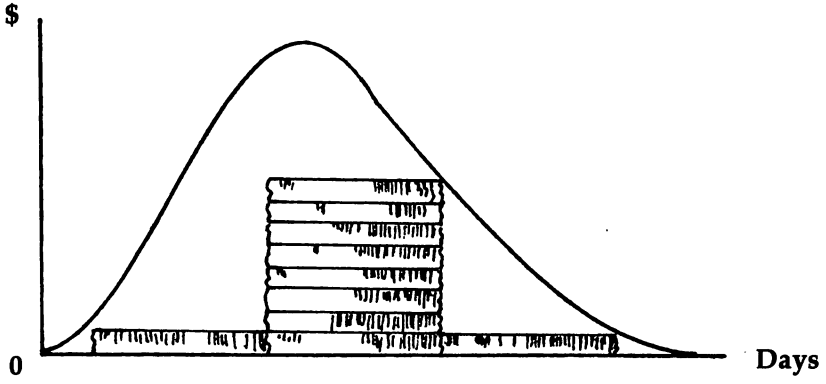
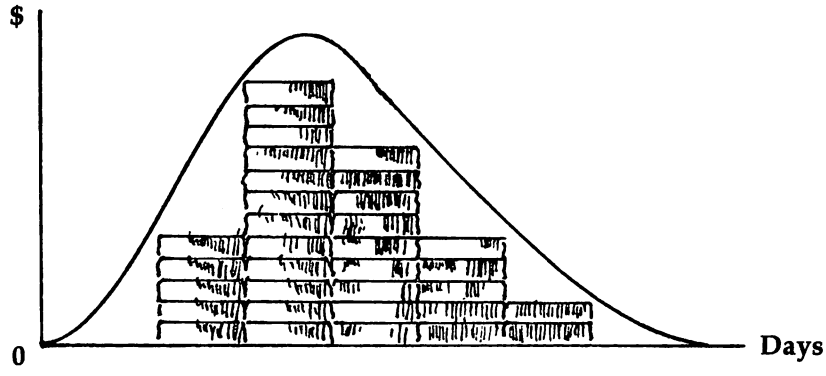


Figure 7-5, for example, shows a plot of daily sales versus number of days. The region under the curve is therefore *total* sales. One way to compute this value might be to stuff the curve with silver dollars. The number of coins required to fill the curve would be an estimate of sales volume. But such an approximation is not very accurate, since a lot of unfilled space remains under the curve. You can lessen this error by using smaller coins—for instance, dimes—instead of silver dollars. The smaller the coin, the better the estimate, since smaller coins would fill more of the area under the curve. In fact, a coin of infinitely small size would yield a perfect estimate.

Figure 7-6. Computing Total Sales



Stuffing the curve with silver dollars gives a crude estimate of total sales. Too many spaces are unfilled.



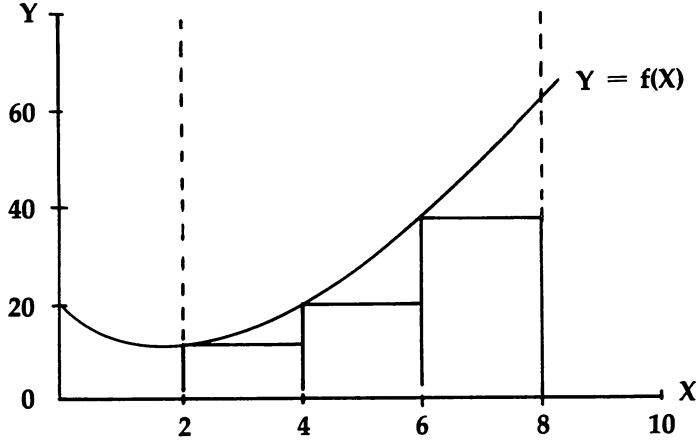
Using dimes gives a better approximation.

This coin-stuffing process is analogous to what the TI does when it computes an integral. Instead of using coins, however, the TI uses rectangles. You tell the computer how many rectangles to use, and the more it uses, the better your approximation will be.

Before running the integration program, first make sure that line 210 reflects the equation of the curve that you want to integrate. Then enter the *limits of integration*, or the lower and upper bounds that delineate the area you're interested in. That's all there is to it.

An example may be helpful. Assume that you are working with the curve Y = 20 − 3*Y + X*X. To compute the area bounded by the X axis, the curve, and the vertical lines X = 2 and X = 8, as shown in Figure 7-7, perform the following steps:

Figure 7-7. Area Bounded by a Curve



1. Using correct BASIC syntax, make sure your curve is represented by line number 210 of the computer program. In this case, line 210 should read:

   210 Y = 20 − 3*X + X*X

2. Enter 2 and 8 as the lower and upper limits of integration.
3. Finally, enter 3 for the number of trapezoids to use in computing the area.

The TI responds with an integral area of 202 units, but you'll get a better answer if you tell the computer to use a larger number of trapezoids. With 20 trapezoids, for example, you'll get an estimate of 198.001, only 0.001 off the exact value.

Reference: *Mathematics,* Life Science Library, Time Incorporated, New York, 1963.

Program 7-4. Numerical Integration

```
100 REM NUMERICAL INTEGRATION
130 REM INITIALIZE
140 GOSUB 230
150 REM COMPUTE
160 GOSUB 680
170 REM DISPLAY RESULT
180 GOSUB 810
190 END
200 REM FUNCTION
210 Y=3+X^3
220 RETURN
230 REM INITIALIZE
240 REM TITLE
250 GOSUB 310
260 REM HEADING
270 GOSUB 370
280 REM LIMITS OF INTEGRATION
290 GOSUB 500
300 RETURN
310 REM TITLE
320 CALL CLEAR
330 DISPLAY AT(12,9):"Numerical"
340 DISPLAY AT(14,10):"Integration"
350 FOR DELAY=1 TO 500 :: NEXT DELAY
360 RETURN
370 REM HEADING
380 CALL CLEAR
390 DISPLAY AT(1,1):"THIS PROGRAM TALLIES THE"
400 DISPLAY AT(2,1):"DEFINITE INTEGRAL OF A"
410 DISPLAY AT(3,1):"FUNCTION USING THE"
420 DISPLAY AT(4,1):"TRAPEZOIDAL RULE."
430 DISPLAY AT(7,1):"IS YOUR FUNCTION NOW IN"
440 DISPLAY AT(8,1):"LINE 210 OF THE PROGRAM"
450 DISPLAY AT(9,1):"(Y/N) ?"
460 ACCEPT AT(9,9)BEEP VALIDATE("YN","")SIZE(1):S$
470 IF S$="" THEN 460
```

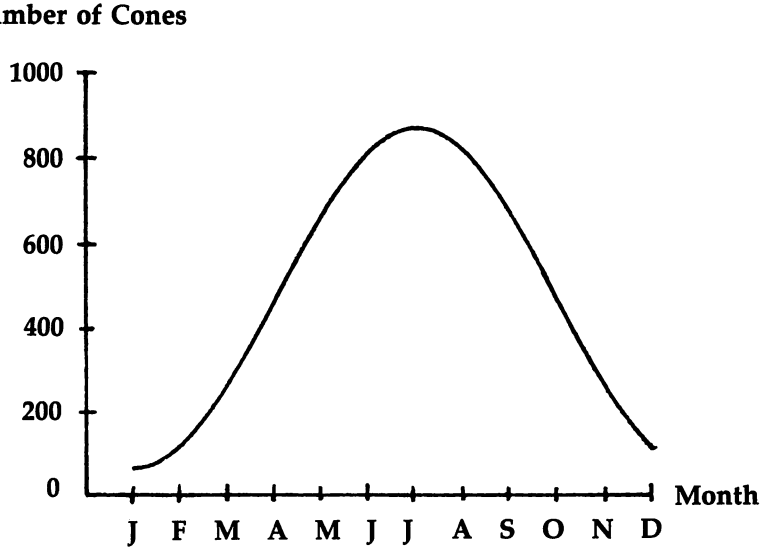
```
480 IF S$="N" THEN DISPLAY AT(23,1):"PLEASE PUT IT
     THERE, AND" :: DISPLAY AT(24,1):"THEN RUN AGA
    IN." :: STOP
490 RETURN
500 REM LIMITS
510 CALL CLEAR
520 DISPLAY AT(1,1):"PLEASE ENTER THE LIMITS OF"
530 DISPLAY AT(2,1):"INTEGRATION."
540 DISPLAY AT(4,1):"LOWER = ?"
550 ACCEPT AT(4,11)BEEP VALIDATE(NUMERIC,""):L$
560 IF L$="" THEN 550 ELSE L=VAL(L$)
570 DISPLAY AT(6,1):"UPPER = ?"
580 ACCEPT AT(6,11)BEEP VALIDATE(NUMERIC,""):U$
590 IF U$="" THEN 580 ELSE U=VAL(U$)
600 IF U<=L THEN DISPLAY AT(23,1):"UPPER LIMIT MUS
    T BE GREATER" :: DISPLAY AT(24,1):"THAN THE LO
    WER !" :: GOTO 580
610 DISPLAY AT(9,1):"HOW MANY INTERVALS WITHIN"
620 DISPLAY AT(10,1):"THESE LIMITS DO YOU WANT TO"
630 DISPLAY AT(11,1):"USE ?"
640 ACCEPT AT(11,7)BEEP VALIDATE(DIGIT,""):N$
650 IF N$="" THEN 640 ELSE N=VAL(N$)
660 IF N=0 THEN 640
670 RETURN
680 REM COMPUTE
690 CALL CLEAR
700 DISPLAY AT(12,1):"COMPUTING ..."
710 REM INTERVAL SIZE
720 DELTA=(U-L)/N
730 SUM=0
740 FOR X=L TO U STEP DELTA
750 GOSUB 210
760 IF X<>L AND X<>U THEN Y=2*Y
770 SUM=SUM+Y
780 NEXT X
790 SUM=DELTA*SUM/2
800 RETURN
810 REM DISPLAY RESULTS
820 IMAGE = ########.###
830 CALL CLEAR
840 CALL HCHAR(1,3,61,28)
850 DISPLAY AT(2,9):"INTEGRATION"
860 CALL HCHAR(3,3,61,28)
870 DISPLAY AT(5,1):"Upper Limit" :: DISPLAY AT(5,
    13):USING 820:U
880 DISPLAY AT(6,1):"Lower Limit" :: DISPLAY AT(6,
    13):USING 820:L
890 DISPLAY AT(9,1):"Number of" :: DISPLAY AT(10,2
    ):"Intervals"
```

```
900 DISPLAY AT(10,13):USING 820:N
910 DISPLAY AT(12,1):"Size of Each" :: DISPLAY AT(
    13,2):"Interval"
920 DISPLAY AT(13,13):USING 820:DELTA
930 DISPLAY AT(16,1):"Integral"
940 DISPLAY AT(16,13):USING 820:SUM
950 CALL HCHAR(23,3,61,28)
960 DISPLAY AT(24,3):"HIT 'ENTER' TO CONTINUE"
970 ACCEPT AT(24,26):Z$
980 RETURN
```

Derivative

Perhaps the only constant in today's world is change. Iron rusts, people age, and the number of ice cream cones sold on Coney Island varies according to the month of the year. But *how* do ice cream cone sales vary over time? To find out, you need to find the *derivative* of the function represented in Figure 7-8.

Figure 7-8. Ice Cream Sales on Coney Island



A derivative measures the rate of change of a function. A function is simply a mapping between two variables, such as ice cream sales and month of the year. In this case, as you can see from the graph, sales climb to a peak during the summer and fall off as autumn begins. This is equivalent to saying that the derivative is positive from January through July (as sales rise) and negative from August through December (as sales fall).

The TI program calculates the derivative of a function and lets you measure a variable's rate of change at any point in time. To find the rate of change in ice cream sales in June, for example, follow these steps:

1. Make line number 220 of the program equal to the equation for your curve. Be sure to use correct BASIC syntax. In this case line 220 should read:

   220 Y = −250 + 330*Y − 25*Y↑2

2. Tell the TI where on the curve to evaluate the derivative. Since January = 1, February = 2, and so on, you would enter 6 for June.

The TI responds with a value of 30. That means that sales are increasing at a rate of 30 cones per month in June.

Program 7-5. Derivative

```
100 REM DERIVATIVE
130 REM INITIALIZE
140 GOSUB 240
150 REM COMPUTE
160 GOSUB 520
170 REM DISPLAY
180 GOSUB 760
190 IF AN$="Y" THEN 160
200 END
210 REM FUNCTION
220 Y=3+5*X^2
230 RETURN
240 REM INITIALIZE
250 REM TITLE
260 GOSUB 300
270 REM INSTRUCTIONS
280 GOSUB 380
290 RETURN
300 REM TITLE
```

```
310 CALL CLEAR
320 CALL CHAR(33,"0000000182442FF")
330 CALL CHAR(36,"00000804FE040800")
340 DISPLAY AT(11,9):"Lim !Y/!X"
350 DISPLAY AT(13,9):"!X$0"
360 FOR DELAY=1 TO 500 :: NEXT DELAY
370 RETURN
380 REM INSTRUCTIONS
390 CALL CLEAR
400 DISPLAY AT(1,1):"THIS PROGRAM EVALUATES THE"
410 DISPLAY AT(2,1):"DERIVATIVE OF THE FUNCTION"
420 DISPLAY AT(3,1):"Y = f(X), AT ANY VALUE OF X."
430 DISPLAY AT(5,1):"IT DOES THIS BY COMPUTING"
440 DISPLAY AT(6,1):"!Y/!X FOR A TINY VALUE"
450 DISPLAY AT(7,1):"OF !X."
460 DISPLAY AT(10,1):"IS YOUR FUNCTION NOW IN"
470 DISPLAY AT(11,1):"LINE 220 (Y/N) ?"
480 ACCEPT AT(11,18)BEEP VALIDATE("YN","")SIZE(1):
    S$
490 IF S$="" THEN 480
500 IF S$="N" THEN DISPLAY AT(23,1):"PLEASE PUT IT
     THERE." :: STOP
510 RETURN
520 REM COMPUTE
530 REM ENTER VALUE
540 GOSUB 580
550 REM COMPUTE
560 GOSUB 670
570 RETURN
580 REM ENTER VALUE
590 CALL CLEAR
600 DISPLAY AT(1,1):"PLEASE ENTER THE VALUE AT"
610 DISPLAY AT(2,1):"WHICH YOU WANT THE"
620 DISPLAY AT(3,1):"DERIVATIVE EVALUATED."
630 DISPLAY AT(6,1):"X = ?"
640 ACCEPT AT(6,7)BEEP VALIDATE(NUMERIC,""):X$
650 IF X$="" THEN 640 ELSE X0=VAL(X$)
660 RETURN
670 REM COMPUTE
680 REM VALUE OF FUNCTION AT X0
690 X=X0 :: GOSUB 220 :: Y0=Y
700 REM VALUE OF FUNCTION AT X1
710 X=X0+1/100^4
720 GOSUB 220 :: X1=X :: Y1=Y
730 REM DERIVATIVE
740 DERV=(Y1-Y0)/(X1-X0)
750 RETURN
760 REM DISPLAY
770 CALL CLEAR
```

```
780 CALL HCHAR(1,3,61,28)
790 DISPLAY AT(2,10):"DERIVATIVE"
800 CALL HCHAR(3,3,61,28)
810 CALL HCHAR(23,3,61,28)
820 IMAGE = ########.###
830 DISPLAY AT(6,12):"X"
840 DISPLAY AT(6,14):USING 820:X0
850 DISPLAY AT(7,3):"DERIVATIVE"
860 DISPLAY AT(7,14):USING 820:DERV
870 DISPLAY AT(15,1):"WOULD YOU LIKE TO EVALUATE"
880 DISPLAY AT(16,1):"THE DERIVATIVE ONCE"
890 DISPLAY AT(17,1):"MORE (Y/N) ?"
900 ACCEPT AT(17,14)BEEP VALIDATE("YN","")SIZE(1):
    AN$
910 IF AN$="" THEN 900
920 RETURN
```

Index

If you've enjoyed the articles in this book, you'll find the same style and quality in every monthly issue of **COMPUTE!** Magazine. Use this form to order your subscription to **COMPUTE!.**

For Fastest Service,
Call Our **Toll-Free** US Order Line

800-334-0868

In NC call 919-275-9809

COMPUTE!
P.O. Box 5406
Greensboro, NC 27403

My Computer Is:
☐ Commodore 64 ☐ TI-99/4A ☐ Timex/Sinclair ☐ VIC-20 ☐ PET
☐ Radio Shack Color Computer ☐ Apple ☐ Atari ☐ Other _____
☐ Don't yet have one...

☐ $24 One Year US Subscription
☐ $45 Two Year US Subscription
☐ $65 Three Year US Subscription

Subscription rates outside the US:

☐ $30 Canada
☐ $42 Europe, Australia, New Zealand/Air Delivery
☐ $52 Middle East, North Africa, Central America/Air Mail
☐ $72 Elsewhere/Air Mail
☐ $30 International Surface Mail (lengthy, unreliable delivery)

Name _____

Address _____

City _____ State _____ Zip _____

Country _____

Payment must be in US Funds drawn on a US Bank; International Money Order, or charge card.

☐ Payment Enclosed ☐ VISA
☐ MasterCard ☐ American Express
Acct. No. _____ Expires _____

What do financial analyses, statistical techniques, bookkeeping chores, and games have in common?

They can all be handled on your TI computer, and *33 Programs for the TI-99/4A* shows you how.

- "IRA Planner" evaluates your Individual Retirement Account
- "Computer Cash Register" lets your TI keep track of sales and sales tax
- "Frequency Plot" gives graphic meaning to statistical data
- "Numerical Integration" and "Derivative" introduce your computer to the world of calculus
- "Hide Brer Rabbit" offers a fascinating variation on the age-old game of "Hangman"—and teaches spelling at the same time

These and almost 30 other programs are ready to type in and run on any TI-99/4A equipped with TI Extended BASIC. Every routine has been carefully designed for ease of use, and most are easily adaptable to your individual needs.

You'll find *33 Programs for the TI-99/4A* an extensive collection of software. With it, and your computer, you can explore the power beneath your keyboard.

ISBN 0-942386-42-6