FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '94
BASIC PROGRAM SOLUTIONS

```
'1.1
' This program will display the 1994 FHSCC sponsors.
'
PRINT "FHSCC '94 IS SPONSORED BY:": PRINT
FOR I = 1 TO 4
  PRINT "GTEDS  GTEDS  GTEDS  GTEDS  GTEDS"
NEXT I
PRINT
FOR I = 1 TO 4
  PRINT "USF CENTER FOR EXCELLENCE"
NEXT I
PRINT
FOR I = 1 TO 4
  PRINT "FLORIDA DEPARTMENT OF EDUCATION"
NEXT I


'1.2
' This program will determine if an applicant is hired.
'
INPUT "Entrance requirement:"; ENT$
INPUT "Plans to accept or reject offer:"; OFFER$
PRINT "APPLICANT WILL ";
IF ENT$ <> "PASSED" OR OFFER$ <> "ACCEPT" THEN PRINT "NOT ";
PRINT "BE HIRED"


'1.3
' This program will display number of employees.
'
INPUT "Enter current number:"; CURRENT
INPUT "Enter number hiring:"; HIRING
INPUT "Enter number leaving:"; LEAVING
PRINT CURRENT + HIRING - LEAVING; "EMPLOYEES"


'1.4
' This program will total the millions converted.
'
INPUT "Enter number of accounts: "; NUM$
WHILE VAL(NUM$) > -999
  SUM = SUM + VAL(NUM$)
  INPUT "Enter number of accounts:"; NUM$
WEND
IF SUM = INT(SUM) THEN
  PRINT SUM;
ELSE
  PRINT USING "#.# "; SUM;
END IF
PRINT "MILLION ACCOUNTS CONVERTED TO CBSS"
```

```
'1.5
' This program will compute the gross wages earned.
'
INPUT "Enter hours, rate:"; HOURS, RATE
IF HOURS > 40 THEN HOURS = HOURS + (HOURS - 40) * .5
PRINT USING "GROSS WAGES ARE $###.##"; HOURS * RATE


'1.6
' This program will tally the number of accounts sold.
'
DATA 706,95000, 208,54321, 912,99825, 605,88776, 404,90175
FOR I = 1 TO 5: READ AREAC(I), ACCT(I): NEXT I
INPUT "Enter number of area codes:"; NUM
FOR I = 1 TO NUM
  INPUT "Enter area code:"; ACODE
  FOR J = 1 TO 5
    IF AREAC(J) = ACODE THEN SUM = SUM + ACCT(J)
  NEXT J
NEXT I
PRINT "TOTAL NUMBER OF ACCOUNTS BEING SOLD ="; SUM


'1.7
' This program will display the cost to fix error in phase.
'
DATA REQUIREMENTS,DESIGN,CODING,SYSTEM TEST,ACCEPTANCE TEST
DATA MAINTENANCE
DATA 1, 5, 10, 20, 50, 100
FOR I = 1 TO 6: READ PHASES$(I): NEXT I
FOR I = 1 TO 6: READ FACTOR(I): NEXT I
INPUT "Enter cost $:"; COST
INPUT "Enter phase:"; PH$
I = 1
WHILE PH$ <> PHASES$(I): I = I + 1: WEND
C$ = LTRIM$(STR$(COST * FACTOR(I)))
PRINT "COST IS $"; C$;
PRINT " TO FIX PROBLEM IN "; PHASES$(I); " PHASE"


'1.8
' This program will compute the maximum blocksize.
'
INPUT "Enter logical record length: "; LRECL
NUM = INT(23476 / LRECL)
PRINT "BLOCKSIZE ="; LRECL * NUM; "BYTES"
```

```
'1.9
' This program will compute an electric bill.
'
INPUT "Enter kilowatt hours:"; HOURS
IF HOURS < 10 THEN RATE = 4.95 ELSE RATE = 5.65
BILL = RATE * HOURS
BILL = BILL * (1 + .03 + .06)
IF HOURS > 30 THEN BILL = BILL + 25
PRINT "THE CUSTOMER'S BILL IS $";
IF BILL < 100 THEN PRINT USING "##.##"; BILL: END
PRINT USING "###.##"; BILL
```

```
'1.10
' This program will determine if a 5x5 matrix is symmetric
'
DEFINT A-Z
FOR I = 1 TO 5
  PRINT "Enter row:";
  INPUT A(I, 1), A(I, 2), A(I, 3), A(I, 4), A(I, 5)
NEXT I
SYM = -1
FOR I = 1 TO 5
  FOR J = 1 TO 5
    IF A(I, J) <> A(J, I) THEN SYM = 0
  NEXT J
NEXT I
PRINT "MATRIX IS ";
IF NOT SYM THEN PRINT "NOT ";
PRINT "SYMMETRIC"
```

```
'2.1
' This program will simulate NTF's ESP utility.
'
DIM JOB$(20)
INPUT "Enter jobs/CK:"; JOBS$
L = INT((LEN(JOBS$) + 1) / 3)
FOR I = 1 TO L
  JOB$(I) = MID$(JOBS$, I * 3 - 2, 2)
NEXT I
I = 0: LASTCK = 0
WHILE I < L
  I = LASTCK + 1
  WHILE JOB$(I) <> "CK"
    PRINT JOB$(I)
    I = I + 1
  WEND
  PRINT "EVERYTHING OK?": INPUT OK$
  IF OK$ = "N" THEN I = LASTCK ELSE LASTCK = I
WEND


'2.2
' This program will display random letters in random areas.
'
RANDOMIZE TIMER
CH$ = " ": LASTLET$ = " "
WHILE CH$ = " " OR (CH$ >= "A" AND CH$ <= "Z")
  CLS
  IF CH$ <> " " THEN
    LETTER$ = CH$
  ELSE
    LETTER$ = CHR$(65 + INT(RND(3) * 26)): CH$ = LETTER$
  END IF
  LASTLET$ = LETTER$
  WHILE CH$ = LASTLET$
    R = INT(RND(3) * 23) + 1:  C = INT(RND(3) * 79) + 1
    LOCATE R, C: PRINT LETTER$;
    FOR I = 1 TO 500: NEXT I
    A$ = INKEY$: IF A$ <> "" THEN LETTER$ = A$: CH$ = A$
  WEND
WEND
```

```
'2.3
' This program will transliterate Hebrew to English.
'
INPUT "Enter letters:"; ST$
LASTCH$ = " "
FOR I = 1 TO LEN(ST$)
  CH$ = MID$(ST$, I, 1): LET$ = CH$
  IF LASTCH$ = " " THEN
    IF CH$ = "A" THEN
      MD$ = MID$(ST$, I + 1, 1)
      IF MD$ = "L" THEN LET$ = ")" ELSE LET$ = "("
    END IF
    IF MID$(ST$, I, 3) = "HET" THEN LET$ = "CH"
    IF MID$(ST$, I, 2) = "TS" THEN LET$ = "TS"
    TRANS$ = LET$ + TRANS$
  END IF
  LASTCH$ = CH$
NEXT I
PRINT TRANS$
```

```
'2.4
' This program will append a "security digit" to an account.
'
INPUT "Enter account number:"; ACCT$
L = LEN(ACCT$)
IF L <> 7 AND L <> 9 THEN
  PRINT "ERROR - INCORRECT LENGTH": ER = -1
END IF
' Sum the valid digits
FOR I = 1 TO L
  CH$ = MID$(ACCT$, I, 1)
  DIG = ASC(CH$) - ASC("0")
  IF DIG < 0 OR DIG > 9 THEN
    PRINT "ERROR - NUM-NUMERIC": END
  END IF
  SUM = SUM + DIG
NEXT I
' If account is valid, append security digit
IF ER THEN END
PRINT ACCT$;
IF SUM MOD 2 = 0 THEN PRINT "1";  ELSE PRINT "0"
```

```
'2.5
' This program will count the digits used in a book.
'
DEFINT A-Z
INPUT "Enter last page:"; LPAGE
INPUT "Enter M:"; M
FOR I = 2 TO LPAGE
  IF I MOD M > 0 THEN
    PAGE$ = MID$(STR$(I), 2)
    FOR J = 1 TO LEN(PAGE$)
      DIG = VAL(MID$(PAGE$, J, 1))
      A(DIG) = A(DIG) + 1
    NEXT J
  END IF
NEXT I
MIN = 32000
FOR I = 0 TO 9
  PRINT I; "APPEARS"; A(I); "TIMES"
  IF A(I) > MAX THEN MAX = A(I)
  IF A(I) < MIN THEN MIN = A(I)
NEXT I
PRINT
PRINT "DIGIT(S) APPEARING THE MOST:";
FOR I = 0 TO 9
  IF A(I) = MAX THEN PRINT USING "##"; I;
NEXT I: PRINT
PRINT "DIGIT(S) APPEARING THE LEAST:";
FOR I = 0 TO 9
  IF A(I) = MIN THEN PRINT USING "##"; I;
NEXT I
```

```
'2.6
' This program will compute the roots for a quadratic.
'
DEFINT A-Z
INPUT "Enter coefficients A, B, C:"; A, B, C
D = B * B - 4 * A * C
PRINT "THE ROOTS ARE ";
IF D >= 0 THEN
  PRINT "REAL"
  R1 = (-B + INT(SQR(D))) / (2 * A)
  R2 = (-B - INT(SQR(D))) / (2 * A)
  GOSUB RemoveSpace
  IF D > 0 THEN
    PRINT "THE ROOTS ARE "; R1$; " AND "; R2$: END
  ELSE
    PRINT "THE ONLY ROOT IS "; R1$: END
  END IF
END IF
' D < 0   Roots are Complex
PRINT "COMPLEX"
R1 = -B / (2 * A)
R2 = INT(SQR(-D)) / (2 * A)
GOSUB RemoveSpace
PRINT "THE ROOTS ARE "; R1$; " + "; R2$; "I AND ";
PRINT R1$; " - "; R2$; "I"
END
' Subroutine to remove leading space, not negative sign
RemoveSpace:
  R1$ = LTRIM$(STR$(R1)):  R2$ = LTRIM$(STR$(R2))
  RETURN
```

```
'2.7
' This program will generate 5 customer account numbers.
'
DEFINT A-Z
DEFDBL S
INPUT "Enter seed used last:"; S
WHILE I < 15
' -- Add 1 and reverse last 2 digits
  S = S + 1
  CUST$ = MID$(STR$(S), 2): L = LEN(CUST$)
  IF L < 9 THEN CUST$ = STRING$(9 - L, "0") + CUST$
  LAST2$ = MID$(CUST$, 9, 1) + MID$(CUST$, 8, 1)
  CUST$ = LEFT$(CUST$, 2) + LAST2$ + MID$(CUST$, 3, 5)
' -- Calculate check digit
  SUM = 0
  FOR J = 1 TO 9
    DIG = VAL(MID$(CUST$, J, 1))
    SUM = SUM + DIG * (11 - J)
  NEXT J
  CDIG = 11 - (SUM MOD 11)
  IF CDIG = 11 THEN CDIG = 0
  IF CDIG < 10 THEN
    PRINT CUST$; : PRINT USING "#"; CDIG: I = I + 1
  END IF
WEND


'2.8
' This program will compute speed, distance, and time.
'
INPUT "Enter speed, distance:"; S, D
INPUT "Enter time: "; TIM$
IF TIM$ <> "0" THEN
  L = LEN(TIM$)
  TTYPE$ = MID$(TIM$, L, 1)
  IF TTYPE$ <> "C" THEN
    T = VAL(MID$(TIM$, 1, L - 1))
  ELSE
    HH = VAL(MID$(TIM$, 1, 2))
    MM = VAL(MID$(TIM$, 4, 2))
    T = HH + MM / 60
  END IF
  IF TTYPE$ = "M" THEN T = T / 60
END IF
IF S = 0 THEN
  PRINT USING "SPEED = ###.#"; D / T; : PRINT " MPH"
ELSE
  IF D = 0 THEN
    PRINT USING "DISTANCE = ####.#"; S * T; : PRINT " MILES"
  ELSE  ' TIM$ = "0"
    PRINT USING "TIME = #.##"; D / S; : PRINT " HOURS"
  END IF
END IF
```

```
'2.9
' This program will compute the response time.
'
INPUT "Enter reported date:"; RDATE$
INPUT "Enter reported time:"; RTIME$
INPUT "Enter cleared date:"; CDATE$
INPUT "Enter cleared time:"; CTIME$
RDAY = VAL(MID$(RDATE$, 4, 2))
CDAY = VAL(MID$(CDATE$, 4, 2))
RHOUR = VAL(MID$(RTIME$, 1, 2))
RMIN = VAL(MID$(RTIME$, 4, 2))
CHOUR = VAL(MID$(CTIME$, 1, 2))
CMIN = VAL(MID$(CTIME$, 4, 2))

IF RHOUR < 8 THEN RHOUR = 8: RMIN = 0
IF CHOUR < 8 THEN CHOUR = 8: CMIN = 0
IF CHOUR >= 17 THEN CHOUR = 17: CMIN = 0
IF RHOUR >= 17 THEN RHOUR = 17: RMIN = 0

RES = (CDAY - RDAY) * 9 * 60
RES = RES + (CHOUR - RHOUR) * 60 + (CMIN - RMIN)

PRINT "RESPONSE TIME WAS"; RES; "MINUTES"
```

```
'2.10
' This program will display the discounts for calling plans.
'
INPUT "Enter originating number:"; ORIGNUM$
INPUT "Enter number called:"; TONUM$
INPUT "Handicapped person?:"; HANDICAP$
INPUT "Enter length of call:"; CALLLEN
INPUT "Enter cost of call $:"; COST
ORIGAREA$ = LEFT$(ORIGNUM$, 3)
TOAREA$ = LEFT$(TONUM$, 3)
DIFFAREA = (ORIGAREA$ <> TOAREA$)
PLANA = 9999: PLANB = 9999: PLANC = 9999
IF (CALLLEN >= 5!) AND DIFFAREA THEN
  PLANA = COST * .85
  PCOST = PLANA: P$ = "A": GOSUB DisplayPlan
END IF
IF HANDICAP$ = "YES" THEN
  PLANB = COST * .9
  PCOST = PLANB: P$ = "B": GOSUB DisplayPlan
END IF
IF (TOAREA$ = "407") AND DIFFAREA AND (CALLEN < 3.5) THEN
  PLANC = COST * .8775
  PCOST = PLANC: P$ = "C": GOSUB DisplayPlan
END IF
IF P$ = "" THEN
  PRINT "THIS PERSON DOES NOT QUALIFY FOR ANY PLANS"
ELSE
  PRINT "THIS PERSON WOULD RECEIVE PLAN ";
  IF PLANA < PLANB AND PLANA < PLANC THEN PRINT "A": END
  IF PLANB < PLANA AND PLANB < PLANC THEN PRINT "B": END
  PRINT "C"
END IF
END
' Subroutine to display plan charges
DisplayPlan:
  PRINT "THE PLAN "; P$; " CHARGE WOULD BE $";
  IF PCOST < 10 THEN
    PRINT USING "#.##"; PCOST
  ELSE
    PRINT USING "##.##"; PCOST
  END IF
  RETURN
```

```
'3.1
' This program will convert transliterated English to Greek
'
DIM NAME$(24), VALUE(24)
DATA ALPHA,BETA,GAMMA,DELTA,EPSILON,ZETA,-TA,IOTA,KAPPA
DATA LAMBDA,MU,NU,XI,-MICRON,PI,RHO,SIGMA,TAU,UPSILON
DATA PHI,CHI,PSI,OMEGA,THETA
DATA 1,2,3,4,5,7,8,10,20,30,40,50,60,70,80
DATA 100,200,300,400,500,600,700,800,9
FOR I = 1 TO 24: READ NAME$(I): NEXT I
FOR I = 1 TO 24: READ VALUE(I): NEXT I
INPUT "Enter transliteration:"; TRANS$
I = 1
WHILE I <= LEN(TRANS$)
  CH$ = MID$(TRANS$, I, 2)
  DOUB = (CH$ = "TH") OR (CH$ = "PH")
  DOUB = DOUB OR (CH$ = "CH") OR (CH$ = "PS")
  IF DOUB THEN INC = 2 ELSE INC = 1
  J = 1
  WHILE MID$(TRANS$, I, INC) <> MID$(NAME$(J), 1, INC)
    J = J + 1
  WEND
  PRINT NAME$(J); " ";
  SUM = SUM + VALUE(J)
  I = I + INC
WEND
PRINT : PRINT "NUMERICAL SUM ="; SUM



'3.2
' This program will move a taxi in a grid.
'
SOUTH = 8
INPUT "Enter starting position:"; SLET$, SNUM
NUM = SNUM
SNUMLET = ASC(SLET$) - ASC("A") + 1: NUMLET = SNUMLET
DO UNTIL DIR$ = "Q"
  INPUT "Enter direction:"; DIR$
  OCL = 0: TOOFAR = 0
  SELECT CASE DIR$
    CASE "N"
      IF NUM = 1 THEN
        OCL = -1
      ELSE
        IF SNUM - 2 = NUM THEN TOOFAR = -1 ELSE NUM = NUM - 1
      END IF
    CASE "S"
      IF NUM = SOUTH THEN
        OCL = -1
      ELSE
        IF SNUM + 2 = NUM THEN TOOFAR = -1 ELSE NUM = NUM + 1
      END IF
    CASE "W"
      IF NUMLET = 1 THEN
```

```
        OCL = -1
      ELSE
        IF SNUMLET - 2 = NUMLET THEN
          TOOFAR = -1
        ELSE
          NUMLET = NUMLET - 1
        END IF
      END IF
    CASE "E"
      IF NUMLET = 26 THEN
        OCL = -1
      ELSE
        IF SNUMLET + 2 = NUMLET THEN
          TOOFAR = -1
        ELSE
          NUMLET = NUMLET + 1
        END IF
      END IF
    END SELECT
'  -- Display error or location
  IF OCL THEN
    PRINT "LOCATION IS OUTSIDE CITY LIMITS"
  ELSE
    IF TOOFAR THEN
      PRINT "LOCATION IS TOO FAR ";
      SELECT CASE DIR$
        CASE "N":  PRINT "NORTH"
        CASE "S":  PRINT "SOUTH"
        CASE "W":  PRINT "WEST"
        CASE "E":  PRINT "EAST"
      END SELECT
    ELSE
      IF DIR$ <> "Q" THEN
        PRINT "TAXI LOCATION IS ";
        PRINT CHR$(NUMLET + 64); ","; LTRIM$(STR$(NUM))
      END IF
    END IF
  END IF
LOOP
```

```
'3.3
' This program will display anagrams.
'
INPUT "Enter number of words:"; NUM
FOR I = 1 TO NUM
  INPUT "Enter word:"; W$(I)
NEXT I
' -- Sort words in ascending order
FOR I = 1 TO NUM - 1
  FOR J = I + 1 TO NUM
    IF W$(I) > W$(J) THEN SWAP W$(I), W$(J)
  NEXT J
NEXT I
' -- Sort letters within word and store in W2$()
FOR I = 1 TO NUM
  L = LEN(W$(I))
  FOR J = 1 TO L
    SORTW$(J) = MID$(W$(I), J, 1)
  NEXT J
  FOR J = 1 TO L - 1
    FOR K = J + 1 TO L
      IF SORTW$(J) > SORTW$(K) THEN SWAP SORTW$(J), SORTW$(K)
    NEXT K
  NEXT J
  FOR J = 1 TO L: W2$(I) = W2$(I) + SORTW$(J): NEXT J
NEXT I
' -- Compare every pair of sorted words for a match
FOR I = 1 TO NUM - 1
  FOR J = I + 1 TO NUM
    IF W2$(I) = W2$(J) THEN
      TOT = TOT + 1
      IF TOT = 1 THEN PRINT "ANAGRAMS: ";
      IF TOT > 1 THEN PRINT "           ";
      PRINT W$(I); ", "; W$(J)
    END IF
  NEXT J
NEXT I
IF TOT = 0 THEN PRINT "NO ANAGRAMS IN LIST"
```

```
'3.4
' This program will place money in envelopes.
'
INPUT "Enter amount of money:"; MONEY
INC = INT(MONEY / 2)
FOR A = 1 TO INC - 2
  FOR B = A + 1 TO INC - 1
    FOR C = B + 1 TO INC
'     { -- D will contain the largest amount to disperse }
      D = MONEY - A - B - C
      IF (A < B) AND (B < C) AND (C < D) THEN
'       { -- (D - A) dollars are dispersed to make  }
'       { --         A=B, B=C, C=D, and D=A         }
        PRINT "TAKE ";
        PRINT LTRIM$(STR$(A)); " "; LTRIM$(STR$(B)); " ";
        PRINT LTRIM$(STR$(C)); " "; LTRIM$(STR$(D));
        PRINT " AND DISPERSE"; D - A; "DOLLARS TO MAKE ";
        PRINT LTRIM$(STR$(B)); " "; LTRIM$(STR$(C)); " ";
        PRINT LTRIM$(STR$(D)); " "; LTRIM$(STR$(A))
        TOTAL = TOTAL + 1
      END IF
    NEXT C
  NEXT B
NEXT A
PRINT "TOTAL NUMBER OF SOLUTIONS ="; TOTAL
```

```
'3.5
' This program will convert Gregorian and Julian dates.
'
DIM MONTH(12)
DATA 31,28,31,30,31,30,31,31,30,31,30,31
FOR I = 1 TO 12: READ MONTH(I): NEXT I
INPUT "Enter Julian or Gregorian:"; DTYPE$
INPUT "Enter date:"; DTE$
IF DTYPE$ = "GREGORIAN" THEN
' Convert Gregorian to Julian
  M = VAL(LEFT$(DTE$, 2))
  D = VAL(MID$(DTE$, 4, 2))
  YY$ = MID$(DTE$, 7, 2)
  Y = VAL(YY$)
  DAYS = D
  FOR I = 1 TO M - 1: DAYS = DAYS + MONTH(I): NEXT I
  IF (Y MOD 4 = 0) AND (M > 2) THEN DAYS = DAYS + 1
  PRINT "JULIAN DATE = "; YY$;
  IF DAYS < 100 THEN PRINT "0";
  IF DAYS < 10 THEN PRINT "0";
  PRINT LTRIM$(STR$(DAYS))
ELSE
' Convert Julian to Gregorian
  YY$ = LEFT$(DTE$, 2)
  Y = VAL(YY$)
  D = VAL(MID$(DTE$, 3, 3))
  M = 1
  IF Y MOD 4 = 0 THEN MONTH(2) = 29
  WHILE D > MONTH(M)
    D = D - MONTH(M)
    M = M + 1
  WEND
  PRINT "GREGORIAN DATE = ";
  PRINT RIGHT$(STR$(100 + M), 2); "/";
  PRINT RIGHT$(STR$(100 + D), 2); "/";
  PRINT YY$
END IF
```

```
'3.6
' This program will convert a number from one base to another.
'
INPUT "Enter base of first number:"; BASE1
INPUT "Enter number:"; NUM1$
INPUT "Enter base of output:"; BASE2
' Convert Num1$ to base 10 number Num1V
FOR I = 1 TO LEN(NUM1$)
  CH$ = MID$(NUM1$, I, 1)
  DIGIT = ASC(CH$) - ASC("0")
  IF DIGIT > 9 THEN DIGIT = DIGIT - 7
  POWER = 1
  FOR J = 1 TO LEN(NUM1$) - I
    POWER = POWER * BASE1
  NEXT J
  NUM1V = NUM1V + DIGIT * POWER
NEXT I
' Convert Num1V to Base2 number
J = INT(LOG(NUM1V) / LOG(BASE2))
FOR I = J TO 0 STEP -1
  POWER = 1
  FOR K = 1 TO I: POWER = POWER * BASE2: NEXT K
  X = INT(NUM1V / POWER)
  NUMOUT$ = MID$("0123456789ABCDEF", X + 1, 1) + NUMOUT$
  NUM1V = NUM1V - X * POWER
NEXT I
PRINT NUMOUT$
```

```
'3.7
' This program will SHELL sort numbers generated.
'
DIM X(-1093 TO 8000)
NUM = 8000: MAX = 7
INPUT "Enter seed X(0):"; X(0)
POW = 1
FOR I = 1 TO 20: POW = POW * 2: NEXT I
FOR I = 1 TO 8000
  Q = INT((69069 * X(I - 1)) / POW)
  X(I) = 69069 * X(I - 1) - POW * Q
NEXT I
' Shell sort routine
INCR(MAX) = 1
FOR I = MAX - 1 TO 1 STEP -1
  INCR(I) = 3 * INCR(I + 1) + 1
NEXT I
FOR I = 1 TO MAX
  INCREMENT = INCR(I)
  FOR J = 1 TO INCREMENT
    LAST = INCREMENT + J
    WHILE LAST <= NUM
      P = LAST
      T = X(P)
      X(1 - INCREMENT) = T
      WHILE T < X(P - INCREMENT)
        X(P) = X(P - INCREMENT)
        P = P - INCREMENT
      WEND
      X(P) = T
      LAST = LAST + INCREMENT
    WEND
  NEXT J
NEXT I
' Display every 1000th number in ascending order
FOR I = 1 TO INT(NUM / 1000)
  PRINT USING "####"; I * 1000;
  PRINT "TH NUMBER ="; X(I * 1000)
NEXT I
```

```
'3.8
' This program will compute the volume of a sphere using PI.
'
DEFINT A-Z
PI1$ = "31415926535897932384626433832795028884"
PI2$ = "19716939937510582097494459230781640 62"
PI3$ = "86208998628034825342117067982148086 51"
PI$ = PI1$ + PI2$ + PI3$
DIM PROD(120)
INPUT "Enter N:"; N
INPUT "Enter radius:"; RADIUS
' Assign digits of PI to Array PI()
L = LEN(PI$)
FOR I = 1 TO L
  PROD(I) = VAL(MID$(PI$, L - I + 1, 1))
NEXT I
'
FOR I = 1 TO 3: A(I) = RADIUS: NEXT I
A(4) = 4
' Multiply PI by Radius (3 times) then by 4
FOR I = 1 TO 4
  FOR J = 1 TO L
    PROD(J) = PROD(J) * A(I) + C
    C = INT(PROD(J) / 10)
    PROD(J) = PROD(J) - C * 10
  NEXT J
  WHILE C > 0
    CC = INT(C / 10)
    L = L + 1
    PROD(L) = C - CC * 10
    C = CC
  WEND
NEXT I
' Divide the product by 3
FOR I = L TO 1 STEP -1
  PR = PROD(I) + R * 10
  PROD(I) = INT(PR / 3)
  R = PR - PROD(I) * 3
NEXT I
IF PROD(L) = 0 THEN L = L - 1
' Display the Volume with the decimal point.
FOR I = L TO 111 - N STEP -1
  IF I = 110 THEN PRINT ".";
  PRINT USING "#"; PROD(I);
NEXT I
```

```
'3.9
' This program will display the barcode of an address.
'
DATA 7,4,2,1,0
FOR I = 1 TO 5: READ VALUE(I): NEXT I
INPUT "Enter address 1:"; ADDR1$
INPUT "Enter address 2:"; ADDR2$
' Extract Zip+4 or Zip from 2nd line of address
L = LEN(ADDR2$)
I = L
WHILE MID$(ADDR2$, I, 1) <> " ": I = I - 1: WEND
IF L - I = 10 THEN
  BARCODE$ = MID$(ADDR2$, I + 1, 5) + MID$(ADDR2$, L - 3, 4)
ELSE
  BARCODE$ = MID$(ADDR2$, L - 4, 5)
END IF
' Extact possible Zip+4 and/or next 2 Delivery points
IF MID$(ADDR1$, 1, 8) = "P.O. BOX" THEN
  L = LEN(ADDR1$)
  I = L
  WHILE MID$(ADDR1$, I, 1) <> " ": I = I - 1: WEND
  FOR J = 1 TO 4 - (L - I): ZIP4$ = ZIP4$ + "0": NEXT J
  ZIP4$ = ZIP4$ + MID$(ADDR1$, I + 1, L - I)
  DPOINT$ = MID$(ZIP4$, 3, 2)
ELSE
  ZIP4$ = "0000"
  ADDR1$ = "0" + ADDR1$
  P = INSTR(1, ADDR1$, " ")
  DPOINT$ = MID$(ADDR1$, P - 2, 2)
END IF
'
IF LEN(BARCODE$) = 5 THEN BARCODE$ = BARCODE$ + ZIP4$
BARCODE$ = BARCODE$ + DPOINT$
' Calculate Check Digit for 12-digit Barcode and display
FOR I = 1 TO 11
  SUM = SUM + VAL(MID$(BARCODE$, I, 1))
NEXT I
CHECKDIG = 10 - (SUM MOD 10)
IF CHECKDIG = 10 THEN CHECKDIG = 0
BARCODE$ = BARCODE$ + CHR$(CHECKDIG + 48)
PRINT SPACE$(12); "DELIVERY POINT BAR CODE = "; BARCODE$
PRINT
' Display Fram bars and encoded Barcode
PRINT "!";
FOR I = 1 TO 12
  DIG = VAL(MID$(BARCODE$, I, 1))
  NUMBARS = 0
  IF DIG = 0 THEN DIG = 11   ' Exception for 0 = 7 + 4
  FOR J = 1 TO 5
    IF (DIG >= VALUE(J)) AND (NUMBARS < 2) THEN
      PRINT "!";
      DIG = DIG - VALUE(J)
      NUMBARS = NUMBARS + 1
    ELSE
      PRINT " ";
```

```
    END IF
  NEXT J
NEXT I
PRINT "!"
FOR I = 1 TO 62: PRINT "!"; : NEXT I



'3.10
' This program produces a 3 x 3 magic square.
'
INPUT "Enter first number:"; FIRSTNUM
INPUT "Enter increment:"; INC
INPUT "Enter number:"; NUM1
INPUT "Enter row, col:"; ROW, COL
POS1 = (ROW - 1) * 3 + COL
INPUT "Enter number:"; NUM2
INPUT "Enter row, col:"; ROW, COL
POS2 = (ROW - 1) * 3 + COL
NUMBER = 7
FOR I = 1 TO NUMBER + 2
  NUM = FIRSTNUM + (I - 1) * INC
  SUM = SUM + NUM
  IF NUM <> NUM1 AND NUM <> NUM2 THEN J = J + 1: S(J) = NUM
NEXT I
MNUM = SUM / 3
' Permute 7 numbers in 3x3 array
FOR N7 = 1 TO 7: H = 6: GOSUB ShiftNums
 FOR N6 = 1 TO 6: H = 5: GOSUB ShiftNums
  FOR N5 = 1 TO 5: H = 4: GOSUB ShiftNums
   FOR N4 = 1 TO 4: H = 3: GOSUB ShiftNums
    FOR N3 = 1 TO 3: H = 2: GOSUB ShiftNums
     FOR N2 = 1 TO 2: J = 0
      FOR I = 1 TO 9
'       Place 2 entered numbers in correct positions
        IF I = POS1 THEN
          SS(I) = NUM1
        ELSE
          IF I = POS2 THEN
            SS(I) = NUM2
          ELSE
            J = J + 1: SS(I) = S(J)
          END IF
        END IF
      NEXT I
      MAGICN = -1
'     Check if row elements sum to Magic Number
      FOR J = 0 TO 2
        SUM = SS(J * 3 + 1) + SS(J * 3 + 2) + SS(J * 3 + 3)
        IF SUM <> MNUM THEN MAGICN = 0
      NEXT J
'     Check if column elements sum to Magic Number
      FOR J = 1 TO 3
        IF SS(J) + SS(J + 3) + SS(J + 6) <> MNUM THEN MAGICN = 0
      NEXT J
```

```
'     Check if diagonal elements sum to Magic Number
      IF MAGICN THEN
        IF (SS(1) + SS(5) + SS(9) = MNUM) THEN
          IF (SS(3) + SS(5) + SS(7) = MNUM) THEN
            FOR J = 0 TO 2
              FOR K = 1 TO 3
                PRINT USING "###"; SS(J * 3 + K);
              NEXT K: PRINT
            NEXT J
            PRINT
            PRINT "MAGIC NUMBER ="; MNUM: END
          END IF
        END IF
      END IF
      SWAP S(NUMBER), S(NUMBER - 1)
     NEXT N2
    NEXT N3
   NEXT N4
  NEXT N5
 NEXT N6
NEXT N7: END
' Subroutine to shift numbers in array
ShiftNums:
  TEMP = S(NUMBER - H)
  FOR J = NUMBER - H TO NUMBER - 1
    S(J) = S(J + 1)
  NEXT J
  S(NUMBER) = TEMP
  RETURN
```