FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '93
BASIC PROGRAM SOLUTIONS

```
'1.1
' This program displays six lines with "GTEDS".
'
FOR I = 1 TO 6
  FOR J = 1 TO 7 - I
    PRINT "GTEDS"; SPACE$(I);
  NEXT J
  PRINT
NEXT I
```

```
'1.2
' This program displays the number of programmers placed.
'
INPUT "Enter N:"; N
INPUT "Enter M:"; M
PRINT N * 15 - M; "PROGRAMMERS"
```

```
'1.3
' This program will format the number N million with commas.
'
INPUT "Enter N:"; N
PRINT USING "##,###,### ACCESS LINES"; N * 1000000!
```

```
'1.4
' This program will total the # of students on 5 USF campuses.
'
DATA Tampa,St. Petersburg,Fort Myers,Lakeland,Sarasota
FOR I = 1 TO 5
  READ CAMPUS$
  PRINT "Enter # at "; CAMPUS$; ":";
  INPUT NUM
  TOTAL = TOTAL + NUM
NEXT I
PRINT TOTAL; "STUDENTS"
```

```
'1.5
' This program will determine if person qualifies for ISOP.
'
INPUT "Enter name:"; NAME$
INPUT "Enter level:"; LEVEL
INPUT "Enter desire:"; DESIRE$
PRINT NAME$; " IS ";
IF (LEVEL < 5) OR (DESIRE$ = "NO") THEN PRINT "NOT ";
PRINT "A POSSIBLE CANDIDATE FOR ISOP"
```

```
'1.6
' This program will display preferred skills for curriculum.
'
INPUT "Enter curriculum:"; CURR$
IF CURR$ = "MVS/COBOL" THEN
  PRINT "COBOL"
  PRINT "JCL"
  PRINT "MVS/ESA"
  PRINT "TSO/ISPF"
  PRINT "VSAM"
  PRINT "ANSI SQL"
  PRINT "DB2"
  PRINT "IMS"
ELSE
  PRINT "C"
  PRINT "UNIX"
  PRINT "ANSI SQL"
  PRINT "OSF/MOTIF"
  PRINT "SHELL PROGRAMMING"
END IF


'1.7
' This program will print the first N letters of alphabet.
'
INPUT "Enter N:"; N
FOR I = 1 TO N
  PRINT CHR$(64 + I);
NEXT I


'1.8
' This program will calculate the increase in salary.
'
INPUT "Enter salary:"; SALARY
INPUT "Enter rating:"; LEVEL$
SELECT CASE LEVEL$
  CASE "EXCELLENT":     INCREASE = SALARY * .1
  CASE "ABOVE AVERAGE": INCREASE = SALARY * .07
  CASE "GOOD":          INCREASE = SALARY * .05
END SELECT
PRINT USING "NEW SALARY = $#####.##"; SALARY + INCREASE


'1.9
' This program will display a Service Order
'
DATA INSTALL,CHANGE,RECORDS,OUT,FROM,TO
INPUT "Enter order: "; ORDER$
CH$ = LEFT$(ORDER$, 1)
IF LEN(ORDER$) > 1 THEN PRINT CH$: END
READ A$
WHILE LEFT$(A$, 1) <> CH$: READ A$: WEND
PRINT A$
```

```
'1.10
' This program will compute a GPA for 5 classes.
'
NUM = 5
FOR I = 1 TO 5
  INPUT "Enter grade:"; G$
  SELECT CASE G$
    CASE "A": SUM = SUM + 4
    CASE "B": SUM = SUM + 3
    CASE "C": SUM = SUM + 2
    CASE "D": SUM = SUM + 1
    CASE "W": NUM = NUM - 1
  END SELECT
NEXT I
PRINT USING "GPA = #.###"; SUM / NUM
```

```
'2.1
' This program will randomly generate #s between X and Y.
'
RANDOMIZE TIMER
INPUT "Enter N:"; N
INPUT "Enter X, Y:"; X, Y
IF X < Y THEN MIN = X: MAX = Y ELSE MIN = Y: MAX = X
FOR I = 1 TO N
  X = INT(RND(3) * (MAX - MIN + 1)) + MIN
  IF X < 0 THEN PRINT " ";
  PRINT STR$(X);
NEXT I
```

```
'2.2
' This program will sort names according to their title.
'
DATA P,PA,SA,SE,SSE,ASE,SASE
FOR I = 1 TO 7: READ TITLES$(I): NEXT I
INPUT "Enter N:"; N
FOR I = 1 TO N
  INPUT "Enter name:"; NAM$(I)
  INPUT "Enter title:"; TITLE$
  NAM$(I) = NAM$(I) + " - " + TITLE$
  J = 1
  WHILE TITLES$(J) <> TITLE$: J = J + 1: WEND
  L(I) = J
NEXT I
FOR I = 1 TO N - 1
  FOR J = I + 1 TO N
    IF L(I) <= L(J) OR (L(I) = L(J) AND NAM$(I) > NAM$(J)) THEN
      SWAP NAM$(I), NAM$(J)
      SWAP L(I), L(J)
    END IF
  NEXT J
NEXT I
FOR I = 1 TO N: PRINT NAM$(I): NEXT I
```

```
'2.3
' This program will foramt a COBOL declaration.
'
DIM FIELD$(15)
I = 0
WHILE (FIELD$(I) > "") OR (I = 0)
  I = I + 1
  INPUT "Enter field:"; FIELD$(I)
WEND
FOR J = 1 TO I - 1
  LEVEL$ = MID$(FIELD$(J), 1, 2)
  IF LEVEL$ = "01" THEN
    INC = 0
  ELSE
    IF LEVEL$ > PREVLEVEL$ THEN INC = INC + 4
    IF LEVEL$ < PREVLEVEL$ THEN INC = INC - 4
  END IF
  PRINT SPACE$(INC);
  PRINT FIELD$(J)
  PREVLEVEL$ = LEVEL$
NEXT J
```

```
'2.4
' This program will translate a word and calculate blocks.
'
INPUT "Enter word:"; WORD$
NUM$ = ""
FOR I = 1 TO LEN(WORD$)
  NUM = ASC(MID$(WORD$, I, 1)) - ASC("A") + 1
  NUM$ = NUM$ + MID$(STR$(NUM), 2)
NEXT I
PRINT "NUMBER = "; NUM$
BLOCKS = 1
LASTDIGIT = VAL(MID$(NUM$, 1, 1))
FOR I = 2 TO LEN(NUM$)
  DIGIT = VAL(MID$(NUM$, I, 1))
  IF DIGIT MOD 2 <> LASTDIGIT MOD 2 THEN BLOCKS = BLOCKS + 1
  LASTDIGIT = DIGIT
NEXT I
PRINT "BLOCKS ="; BLOCKS
```

```
'2.5
' This program will display N formatted telephone #s.
'
INPUT "Enter N:"; N
FOR I = 1 TO N
  INPUT "Enter #:"; NUM$(I)
NEXT I
TOTAL = 1:  NUM$(I + 1) = SPACE$(10)
FOR I = 1 TO N
  NPA$ = MID$(NUM$(I), 1, 3)
  NXX$ = MID$(NUM$(I), 4, 3)
  LIN$ = MID$(NUM$(I), 7, 4)
  PRINT NPA$; "-"; NXX$; "-"; LIN$;
  NEXTNPA$ = MID$(NUM$(I + 1), 1, 3)
  NEXTNXX$ = MID$(NUM$(I + 1), 4, 3)
  IF NPA$ <> NEXTNPA$ THEN
    PRINT "  TOTAL FOR NPA OF "; NPA$; " ="; TOTAL
    PRINT : TOTAL = 1
  ELSE
    TOTAL = TOTAL + 1
    IF NXX$ <> NEXTNXX$ THEN PRINT
  END IF
  PRINT
NEXT I
```

```
'2.6
' This program will calculate product bought minus coupons.
'
WHILE PROD$(I) <> "9"
  I = I + 1
  INPUT "Enter product:"; PROD$(I)
  IF PROD$(I) <> "9" THEN INPUT "Enter price:"; PRIC(I)
WEND
NUMPROD = I - 1
PRINT

DO UNTIL COUP$(J) = "9"
  J = J + 1
  INPUT "Enter coupon:"; COUP$(J)
  IF COUP$(J) <> "9" THEN INPUT "Enter discount:"; DISC(J)
LOOP
NUMCOUP = J - 1

FOR I = 1 TO NUMPROD
  MAXDISC = 0
  FOR J = 1 TO NUMCOUP
    IF PROD$(I) = COUP$(J) AND DISC(J) > MAXDISC THEN
      MAXDISC = DISC(J):  IND = J
    END IF
  NEXT J
  TOTAL = TOTAL + PRIC(I) - MAXDISC
  COUP$(IND) = "*"
NEXT I
PRINT : PRINT "TOTAL = $";
IF TOTAL < 10 THEN PRINT USING "#.##"; TOTAL
IF TOTAL >= 10 THEN PRINT USING "##.##"; TOTAL
```

```
'2.7
' This program will display dates in other formats.
'
INPUT "Enter format:"; format$
INPUT "Enter date:"; DAT$
SELECT CASE format$
  CASE "ISO"
    YYYY$ = MID$(DAT$, 1, 4)
    MM$ = MID$(DAT$, 6, 2)
    DD$ = MID$(DAT$, 9, 2)
  CASE "AMERICAN"
    MM$ = MID$(DAT$, 1, 2)
    DD$ = MID$(DAT$, 4, 2)
    YYYY$ = MID$(DAT$, 7, 4)
  CASE "EUROPEAN"
    DD$ = MID$(DAT$, 1, 2)
    MM$ = MID$(DAT$, 4, 2)
    YYYY$ = MID$(DAT$, 7, 4)
END SELECT
IF format$ <> "ISO" THEN
  PRINT "ISO = "; YYYY$; "-"; MM$; "-"; DD$
END IF
IF format$ <> "AMERICAN" THEN
  PRINT "AMERICAN = "; MM$; "-"; DD$; "-"; YYYY$
END IF
IF format$ <> "EUROPEAN" THEN
  PRINT "EUROPEAN = "; DD$; "-"; MM$; "-"; YYYY$
END IF


'2.8
' This program will reverse the words in 1 or 2 sentences.
'
INPUT "Enter sentence:"; SENT$
NUM = 1: WORD$(NUM) = "":  I = 1
WHILE I <= LEN(SENT$)
  CH$ = MID$(SENT$, I, 1)
  IF CH$ = "." THEN
    FOR J = NUM TO 1 STEP -1
      IF J < NUM THEN PRINT " ";
      PRINT WORD$(J);
    NEXT J
    PRINT ".  ";
    NUM = 0:  I = I + 1
  ELSE  ' -- NOT A PERIOD
    IF CH$ <> " " THEN
      WORD$(NUM) = WORD$(NUM) + CH$
    ELSE
       NUM = NUM + 1: WORD$(NUM) = ""
    END IF
  END IF
  I = I + 1
WEND
```

```
'2.9
' This program will print 4 smallest #s in a 4 x 4 matrix.
'
DIM B(16)
FOR I = 1 TO 4
  PRINT USING "Enter row #:"; I;
  INPUT A(I, 1), A(I, 2), A(I, 3), A(I, 4)
NEXT I
FOR I = 1 TO 4
  FOR J = 1 TO 4
    B((I - 1) * 4 + J) = A(I, J)
  NEXT J
NEXT I
'
FOR I = 1 TO 15
  FOR J = I + 1 TO 16
    IF B(I) > B(J) THEN SWAP B(I), B(J)
  NEXT J
NEXT I
'
K = 1:  B(0) = -99
WHILE (NUM < 4) OR (B(K) = B(K - 1))
  ONEDISP = 0
  IF B(K) <> B(K - 1) THEN
    PRINT
    NUM = NUM + 1
    PRINT USING "#"; NUM;
    PRINT ". SMALLEST ="; B(K); "OCCURS AT ";
    FOR I = 1 TO 4
      FOR J = 1 TO 4
        IF B(K) = A(I, J) THEN
          IF ONEDISP THEN PRINT ", ";  ELSE ONEDISP = 1
          PRINT USING "(#"; I; : PRINT USING ",#"; J;
          PRINT ")";
        END IF
      NEXT J
    NEXT I
  END IF
  K = K + 1
WEND
```

```
'2.10
' This program will print # of days between two dates.
'
DATA 31,28,31,30,31,30,31,31,30,31,30,31
DIM MONTH(12): FOR I = 1 TO 12: READ MONTH(I): NEXT I
INPUT "Enter month:"; M
INPUT "Enter day:"; D
INPUT "Enter year:"; Y
' October 25, 1967
FOR I = 1 TO 9
  DAYS2 = DAYS2 + MONTH(I)
NEXT I
DAYS2 = DAYS2 + 25
'
FOR I = 1967 TO Y - 1
  DAYS = DAYS + 365
  IF I MOD 4 = 0 THEN DAYS = DAYS + 1
NEXT I
IF (Y MOD 4 = 0) AND (M > 2) THEN DAYS = DAYS + 1
FOR I = 1 TO M - 1
  DAYS = DAYS + MONTH(I)
NEXT I
DAYS = DAYS + D
PRINT DAYS - DAYS2; "DAYS"
```

```
'3.1
' This program displays GTEDS squares relative to cursor.
' Cursor can be moved up, left, down, right: I, J, K, M.
CLS
R = 5:  C = 5: K$ = " "
WHILE K$ < "1" OR K$ > "4"
  LOCATE R, C: PRINT "#": K$ = ""
  WHILE K$ = "": K$ = INKEY$: WEND
  IF K$ >= "I" AND K$ <= "M" THEN
    LOCATE R, C: PRINT " "
    IF K$ = "I" THEN R = R - 1
    IF K$ = "M" THEN R = R + 1
    IF K$ = "J" THEN C = C - 1
    IF K$ = "K" THEN C = C + 1
  END IF
WEND
X = ASC(K$) - ASC("0")
IF X = 1 THEN A = 1: B = 0
IF X = 2 THEN A = 1: B = -1
IF X = 3 THEN A = -1: B = -1
IF X = 4 THEN A = -1: B = 0
IF (R + 5 * A > 24) OR (R + 5 * A < 1) THEN
  PRINT "OFF THE SCREEN": END
ELSE
  IF (C + 9 * B + 9 > 80) OR (C + 9 * B < 1) THEN
    PRINT "OFF THE SCREEN": END
  ELSE
    LOCATE R + 1 * A, C + 8 * B: PRINT "G T E D S"
    LOCATE R + 2 * A, C + 8 * B: PRINT "T       D"
    LOCATE R + 3 * A, C + 8 * B: PRINT "E  "; X; "  E"
    LOCATE R + 4 * A, C + 8 * B: PRINT "D       T"
    LOCATE R + 5 * A, C + 8 * B: PRINT "S D E T G"
  END IF
END IF
```

```
'3.2
' This program will solve an equation with +,-,*, or /.
'
INPUT "Enter value:"; V1$
INPUT "Enter symbol:"; S1$
INPUT "Enter value:"; V2$
INPUT "Enter symbol:"; S2$
INPUT "Enter value:"; V3$
IF S1$ = "=" THEN
  S1$ = S2$: S2$ = "="
  X$ = V1$:  V1$ = V2$:  V2$ = V3$:  V3$ = X$
END IF
' Equation is now of the form V1 [op] V2 = V3
N1 = VAL(V1$)
N2 = VAL(V2$)
N3 = VAL(V3$)
PRINT "X =";
SELECT CASE S1$
  CASE "+"
    IF V1$ = "X" THEN PRINT N3 - N2
    IF V2$ = "X" THEN PRINT N3 - N1
    IF V3$ = "X" THEN PRINT N1 + N2
  CASE "-"
    IF V1$ = "X" THEN PRINT N3 + N2
    IF V2$ = "X" THEN PRINT N1 - N3
    IF V3$ = "X" THEN PRINT N1 - N2
  CASE "*"
    IF V1$ = "X" THEN PRINT N3 / N2
    IF V2$ = "X" THEN PRINT N3 / N1
    IF V3$ = "X" THEN PRINT N1 * N2
  CASE "/"
    IF V1$ = "X" THEN PRINT N3 * N2
    IF V2$ = "X" THEN PRINT N1 / N3
    IF V3$ = "X" THEN PRINT N1 / N2
END SELECT
```

```
'3.3
' This program prints combinations of digits summing to #.
'
INPUT "Enter digits:"; DIGITS$
INPUT "Enter sum:"; SUM
NEWSUM = INT(SUM / 10) * 8 + (SUM MOD 10)
LAST = LEN(DIGITS$)
FOR I = 1 TO LAST
  DIGIT(I) = VAL(MID$(DIGITS$, I, 1))
NEXT I
'
POWER = 1
FOR I = 1 TO LAST: POWER = POWER * 2: NEXT I
POWER = POWER - 1
'
FOR I = 1 TO POWER
  J = 1
  WHILE (A(J) = 1)
    A(J) = 0: J = J + 1
  WEND
  A(J) = 1
  TOTAL = 0
  FOR J = 1 TO LAST
    IF A(J) = 1 THEN TOTAL = TOTAL + DIGIT(J)
  NEXT J
  ONEPRINT = 0
  IF TOTAL = NEWSUM THEN
    FOR J = 1 TO LAST
      IF A(J) = 1 THEN
        IF ONEPRINT THEN PRINT "+";  ELSE ONEPRINT = 1
        PRINT USING "#"; DIGIT(J);
      END IF
    NEXT J
    PRINT " ="; SUM
  END IF
NEXT I


'3.4
' This program will decompose a large integer into primes.
'
DIM A(80), Q(80)
INPUT "Enter number:"; LONGNUM$
L = LEN(LONGNUM$)
FOR I = 1 TO L
  A(I) = VAL(MID$(LONGNUM$, I, 1))
NEXT I
PRIME = 2: POWER = 0
FIRSTFACTOR = 1: QUOTIENTIS0 = 0
WHILE NOT QUOTIENTIS0
' Check if LongNum (Array A) is divisble by Prime
  NUM = 0
  FOR I = 1 TO L
    NUM = NUM * 10 + A(I)
    Q(I) = INT(NUM / PRIME)
```

```
    NUM = NUM - Q(I) * PRIME
  NEXT I
  IF NUM = 0 THEN
'    Prime divided LongNum
    I = 1
    WHILE (Q(I) = 0) AND (I <= L): I = I + 1: WEND
    QUOTIENTIS0 = (I = L) AND (Q(L) = 1)
    L = L - I + 1
'    Copy Quotient into array A to be divided again
    FOR J = 1 TO L
      A(J) = Q(J + I - 1)
    NEXT J
    POWER = POWER + 1
  ELSE
'    Prime did not divide LongNum
    IF POWER >= 1 THEN GOSUB DisplayFactor
    GOSUB GetNextPrime
  END IF
WEND
GOSUB DisplayFactor: END
' Display Factor
DisplayFactor:
  IF FIRSTFACTOR THEN FIRSTFACTOR = 0 ELSE PRINT " * ";
  PRINT MID$(STR$(PRIME), 2);
  IF POWER > 1 THEN PRINT "^"; MID$(STR$(POWER), 2);
  POWER = 0
  RETURN
' Get next prime
GetNextPrime:
  IF PRIME = 2 THEN PRIME = 3: RETURN
  ISPRIME = 0
  WHILE ISPRIME = 0
    PRIME = PRIME + 2
    ISPRIME = 1
    FOR J = 3 TO INT(SQR(PRIME))
      IF PRIME MOD J = 0 THEN ISPRIME = 0
    NEXT J
  WEND
  RETURN


'3.5
' This program will find words in a 12 x 11 array of letters.
'
DIM A$(12), B$(12)
A$(1) = "DATAADFBAAM": A$(2) = "JARBJCEDFOI"
A$(3) = "REAEEXEVDBC": A$(4) = "JESUSDEERNR"
A$(5) = "FABUUNMIEMO": A$(6) = "LLMNSOIPTKC"
A$(7) = "POQRSITRUOH": A$(8) = "ABUVKWSXPPI"
A$(9) = "SOYZCPULMLP": A$(10) = "CCISABCDOAM"
A$(11) = "AEFGRHIJCRM": A$(12) = "LKLETTEKSID"
' String together the columns instead of rows
FOR I = 1 TO 11
  B$(I) = ""
  FOR J = 1 TO 12
```

```
    B$(I) = B$(I) + MID$(A$(J), I, 1)
  NEXT J
NEXT I
INPUT "Enter word:"; WORD$(1)
L = LEN(WORD$(1))
' Reverse word
WORD$(2) = ""
FOR I = 1 TO L
  WORD$(2) = WORD$(2) + MID$(WORD$(1), L - I + 1, 1)
NEXT I
'
' Find words horizontally, (frontwards and backwards)
'
J = 0
WHILE (COL = 0) AND (J < 2)
  J = J + 1: ROW = 0
  WHILE (ROW < 12) AND (COL = 0)
    ROW = ROW + 1
    COL = INSTR(1, A$(ROW), WORD$(J))
  WEND
WEND
'
IF COL = 0 THEN
  ROW = 0: J = 0
ELSE
  IF J = 1 THEN C1 = COL: C2 = COL + L - 1
  IF J = 2 THEN C1 = COL + L - 1: C2 = COL
  R1 = ROW: R2 = ROW
  GOTO DisplayCoordinates
END IF
'
' Find words vertically, (frontwards and backwards)
'
WHILE (ROW = 0) AND (J < 2)
  J = J + 1: COL = 0
  WHILE (COL < 11) AND (ROW = 0)
    COL = COL + 1
    ROW = INSTR(1, B$(COL), WORD$(J))
  WEND
WEND
IF ROW = 0 THEN END
IF J = 1 THEN R1 = ROW: R2 = ROW + L - 1
IF J = 2 THEN R1 = ROW + L - 1: R2 = ROW
C1 = COL: C2 = COL
'
' Display coordinates
'
DisplayCoordinates:
  PRINT USING "FIRST LETTER: (##"; R1;
  PRINT USING ", ##"; C1; : PRINT ")"
  PRINT USING "LAST LETTER:  (##"; R2;
  PRINT USING ", ##"; C2; : PRINT ")"
```

```
'3.6
' This program will solve two inequality equations.
'
INPUT "Enter equation 1:"; EQ1$
INPUT "Enter logical op:"; OP$
INPUT "Enter equation 2:"; EQ2$
S1$ = MID$(EQ1$, 2, 1)
S2$ = MID$(EQ2$, 2, 1)
N1 = VAL(MID$(EQ1$, 3, 1))
N2 = VAL(MID$(EQ2$, 3, 1))
NOS1 = (S1$ = "<" AND S2$ = ">" AND OP$ = "AND" AND N1 <= N2)
NOS2 = (S1$ = ">" AND S2$ = "<" AND OP$ = "AND" AND N1 >= N2)
IF NOS1 OR NOS2 THEN PRINT "NO SOLUTION": END
ALL1 = (S1$ = "<" AND S2$ = ">" AND OP$ = "OR" AND N1 > N2)
ALL2 = (S1$ = ">" AND S2$ = "<" AND OP$ = "OR" AND N1 < N2)
IF ALL1 OR ALL2 THEN PRINT "ALL INTEGERS": END
IF N < N2 THEN MIN = N1: MAX = N2 ELSE MIN = N2: MAX = N1
' Check for finite solution, and if less than 6 integers
FIN1 = (S1$ = "<" AND S2$ = ">" AND OP$ = "AND" AND N1 > N2)
FIN2 = (S1$ = ">" AND S2$ = "<" AND OP$ = "AND" AND N1 < N2)
IF (FIN1 OR FIN2) THEN
  IF MAX - MIN > 7 THEN
    A = MIN + 1: B = MIN + 3: GOSUB DisplayNumbers
    PRINT "...";
    A = MAX - 3: B = MAX - 1: GOSUB DisplayNumbers: END
  END IF
  A = MIN + 1: B = MAX - 1: GOSUB DisplayNumbers: END
END IF
' Check for infinite # of negative solutions
IF (S1$ = "<" AND S2$ = "<" AND OP$ = "AND") THEN
  PRINT "...";
  A = MIN - 3: B = MIN - 1: GOSUB DisplayNumbers: END
END IF
' Check for infinite # of positive solutions
IF (S1$ = ">" AND S2$ = ">" AND OP$ = "AND") THEN
  A = MAX + 1: B = MAX + 3
  PRINT "...": END
END IF
' Check for infinite # of positive and negative solutions
IN1 = (S1$ = ">" AND S2$ = "<" AND OP$ = "OR" AND N1 > N2)
IN2 = (S1$ = "<" AND S2$ = ">" AND OP$ = "OR" AND N1 < N2)
IF (IN1 OR IN2) THEN
  PRINT "...";
  A = MIN - 3: B = MIN - 1: GOSUB DisplayNumbers
  PRINT "   ";
  A = MAX + 1: B = MAX + 3: GOSUB DisplayNumbers
  PRINT "...";
END IF
END
' Display numbers
DisplayNumbers:
  IF A < 0 THEN PRINT LEFT$(STR$(A), 2);
  IF A >= 0 THEN PRINT USING "#"; A;
  FOR I = A + 1 TO B
    PRINT ",";
```

```
    IF I < 0 THEN PRINT LEFT$(STR$(I), 2);
    IF I >= 0 THEN PRINT USING "#"; I;
  NEXT I
  RETURN



'3.7
' This program will print the sum and product of 2 matrices.
'
BASE$ = "0123456789ABCDEF"
FOR I = 1 TO 2
  FOR J = 1 TO 3
    FOR K = 1 TO 3
      PRINT USING "Enter Mat#"; I; : PRINT " (";
      PRINT USING "#"; J; : PRINT ","; : PRINT USING "#"; K;
      INPUT ")"; NUM$
      L = LEN(NUM$): TENS = 0
      IF L = 2 THEN
        TENS = (INSTR(1, BASE$, MID$(NUM$, 1, 1)) - 1) * 16
      END IF
      ONES = INSTR(1, BASE$, MID$(NUM$, L, 1)) - 1
      MAT(I, J, K) = TENS + ONES
    NEXT K
  NEXT J
NEXT I
' Compute sum
PRINT "SUM =";
FOR I = 1 TO 3
  FOR J = 1 TO 3
    SUM = MAT(1, I, J) + MAT(2, I, J)
    PRINT SPACE$(6 - LEN(HEX$(SUM))); HEX$(SUM);
  NEXT J
  PRINT
  IF I < 3 THEN PRINT SPACE$(5);
NEXT I
PRINT
' Compute product
PRINT "PRODUCT =";
FOR I = 1 TO 3
  FOR J = 1 TO 3
    PROD = 0
    FOR K = 1 TO 3
      PROD = PROD + MAT(1, I, K) * MAT(2, K, J)
    NEXT K
    PRINT SPACE$(6 - LEN(HEX$(PROD))); HEX$(PROD);
  NEXT J
  PRINT
  IF I < 3 THEN PRINT SPACE$(9);
NEXT I
```

```
'3.8
' This program will find three 3-digit primes.
'
DEFINT A-Z
DIM P(200)
NUM = 101: PNUM = 0
WHILE NUM < 999
  SQ = INT(SQR(NUM)): I = 3
  WHILE (I <= SQ) AND (NUM MOD I > 0): I = I + 1: WEND
  IF I > SQ THEN
    N2 = NUM
    D1 = INT(N2 / 100)
    N2 = N2 - D1 * 100
    D2 = INT(N2 / 10)
    D3 = N2 - D2 * 10
    IF NOT (D1 = 0 OR D2 = 0 OR D3 = 0) THEN
      IF NOT (D1 = D2 OR D2 = D3 OR D1 = D3) THEN
        PNUM = PNUM + 1: P(PNUM) = NUM
      END IF
    END IF
  END IF
  NUM = NUM + 2
WEND
FOR I = 1 TO PNUM - 2
  FOR J = I + 1 TO PNUM - 1
    FOR K = J + 1 TO PNUM
      TOT = P(I) + P(J) + P(K)
      IF TOT > 1234 THEN
        P1$ = MID$(STR$(P(I)), 2)
        P2$ = MID$(STR$(P(J)), 2)
        P3$ = MID$(STR$(P(K)), 2)
        PCAT$ = P1$ + P2$ + P3$
        FOR L = 1 TO 9: A(L) = 0: NEXT L: L = 0
        WHILE (L < 9) AND (A(X) < 2)
          L = L + 1
          X = VAL(MID$(PCAT$, L, 1))
          A(X) = A(X) + 1
        WEND
        IF A(X) < 2 THEN
          SUM$ = MID$(STR$(TOT), 2)
          D1 = (MID$(SUM$, 1, 1) < MID$(SUM$, 2, 1))
          D2 = (MID$(SUM$, 2, 1) < MID$(SUM$, 3, 1))
          D3 = (MID$(SUM$, 3, 1) < MID$(SUM$, 4, 1))
          IF D1 AND D2 AND D3 THEN
            PRINT P(I); "+"; P(J); "+"; P(K); "="; TOT
            DISP = DISP + 1: IF DISP = 7 THEN END
          END IF
        END IF
      END IF
    NEXT K
  NEXT J
NEXT I
```

```
'3.9
' This program will produce a binary search tree.
'
DIM A$(8, 256)
DATA 0,15,7,3,1,0,0,0,0,0
FOR I = 0 TO 8: READ COLINC(I): NEXT I
CLS : INPUT "Enter word(s):"; WORDS$
CLS
FOR I = 1 TO LEN(WORDS$)
  CH$ = MID$(WORDS$, I, 1)
  IF CH$ <> " " THEN
    R = 0: C = 1: COL = 40
'   Traverse tree until an empty node exists
    WHILE A$(R, C) <> ""
      IF CH$ <= A$(R, C) THEN
        C = 2 * C - 1: COL = COL - COLINC(R + 1) - 1
      ELSE
        C = 2 * C
        PREVCOL = COL
        COL = COL + COLINC(R + 1) + 1
      END IF
      R = R + 1
    WEND
    A$(R, C) = CH$
'
    LOCATE R + 1, COL
    IF R = 0 THEN
      PRINT CH$;
    ELSE
      IF C MOD 2 = 1 THEN
        PRINT CH$; STRING$(COLINC(R), "-"); "+";
      ELSE
        LOCATE R + 1, PREVCOL
        PRINT "+"; STRING$(COLINC(R), "-"); CH$;
      END IF
    END IF
  END IF
NEXT I
```

```
'3.10
' This program will determine the values F(X) converges.
'
DIM F#(5000)
FOR I = 1 TO 2
  IF I = 1 THEN INC# = .01 ELSE INC# = .1
  DIVERGE = 0: FACTOR# = 1: FOUND = 0
  WHILE (K# < 10) AND NOT FOUND
    K# = K# + INC# / FACTOR#
    X = 1: F#(X) = K#
    IF FACTOR# < 20 THEN ITER = 250 * FACTOR# ELSE ITER = 5000
    WHILE (X < ITER) AND NOT DIVERGE
      X = X + 1
      F#(X) = EXP(LOG(K#) * F#(X - 1))
      DIVERGE = (F#(X) > 9.9)
    WEND
    IF I = 1 THEN
      FX2# = FX1#: FX1# = FX0#: FX0# = F#(X)
      IF (FX2# > FX1#) AND (FX1# < FX0#) THEN
        K# = K# - 2 * INC# / FACTOR#
        IF (FX2# - FX1#) < .0005 THEN FOUND = -1: FX# = FX1#
        FX0# = FX2#: FX1# = FX0#
        FACTOR# = FACTOR# * 2
      END IF
    ELSE
'     Find Maximum point
      IF DIVERGE THEN
        DIVERGE = 0
        K# = K# - INC# / FACTOR#
        IF INC# / FACTOR# < .000005 THEN FOUND = -1
        FACTOR# = FACTOR# * 2
      ELSE
        FX# = F#(X)
      END IF
    END IF
  WEND
  IF I = 1 THEN PRINT "MINIMUM";  ELSE PRINT "MAXIMUM";
  PRINT " VALUE: ";
  IF I = 1 THEN
    PRINT USING "F(X) = #.###"; FX#; : PRINT " OCCURS WHEN ";
    PRINT USING "K = #.###"; K# + INC# / FACTOR#
  ELSE
    PRINT USING "F(X) = #.#"; FX#; :  PRINT "   OCCURS WHEN ";
    PRINT USING "K = #.#####"; K# + INC# / FACTOR#
  END IF
NEXT I
```