

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '85
BASIC PROGRAM SOLUTIONS

```
'1.1
' This program will simulate a Last-In-First-Out stack.
'
INPUT "Enter command:"; A$
WHILE A$ <> "QUIT"
  IF A$ = "ADD" THEN S = S + 1: INPUT "Enter number:"; A(S)
  IF A$ = "TAKE" THEN PRINT A(S): S = S - 1
  INPUT "Enter command:"; A$
WEND
```

```
'1.2
' This program will determine which number was erased.
'
INPUT "Enter N, AV: "; N, AV
FOR I = 1 TO N: S = S + I: NEXT I
FOR I = 1 TO N
  T = S - I
  IF ABS((T / (N - 1)) - AV) < .0001 THEN
    PRINT "NUMBER ERASED WAS"; I: END
  END IF
NEXT I
```

```
'1.3
' This program will print the square root of N.
'
INPUT "Enter N, D: "; N, D: Q = SQR(N)
S = INT(Q / 10 ^ D + .5) * 10 ^ D
A$ = MID$(STR$(S), 2)
FOR I = 1 TO LEN(A$)
  T = T + VAL(MID$(A$, I, 1))
NEXT I
PRINT USING "S=####.####"; S
PRINT "SUM="; LTRIM$(STR$(T))
```

```
'1.4
' This program will simulate a time dial.
'
CLS : Y = 1985
J = 3000      'About one second on a 386-computer
WHILE Y <= 2345
  LOCATE 12, 38: PRINT Y
  IF J > 30 THEN J = J - 30
  FOR K = 1 TO J: NEXT K
  Y = Y + 1
WEND
```

'1.5

' This program will determine # of tennis games and byes.

,

```
INPUT "Enter N: "; N
WHILE N > 1
  G = INT(N / 2)
  IF G * 2 = N THEN B = 0 ELSE B = 1
  R = R + 1
  PRINT USING "ROUND #  ## GAMES"; R; G;
  IF B = 1 THEN PRINT "  1 BYE" ELSE PRINT
  TG = TG + G: BY = BY + B: N = G + B
WEND
PRINT USING "TOTAL      ## GAMES ## BYES"; TG; BY
```

'1.6

' This program will find smallest, largest, and sum of #s.

,

```
INPUT "Enter N, M: "; N, M
IF M > 999 THEN M = 999
IF N < 100 THEN N = 100
FOR I = N TO M
  NUM = I
  H = INT(NUM / 100): NUM = NUM - H * 100
  T = INT(NUM / 10):  U = NUM - T * 10
  IF NOT (H = 0 OR T = 0 OR U = 0 OR H = T OR T = U OR H = U) THEN
    S = S + I: L = I: IF S = I THEN PRINT "SMALLEST ="; S
  END IF
NEXT I
PRINT "LARGEST ="; L
PRINT "SUM ="; S
```

'1.7

' This program will print a bill for Bob's Cycle shop.

,

```
INPUT "Enter Customer name: "; N$
INPUT "Enter part#: "; P$
INPUT "Enter labor time: "; LT
WHILE P$ <> A$: READ A$, B$, C: WEND
' Print Invoice
PRINT
PRINT "CUSTOMER NAME: "; N$
PRINT "PART #: "; P$
PRINT "DESCRIPTION: "; B$
PRINT USING "PART COST:  ###.##"; C
LC = LT * 10
PRINT USING "LABOR COST: ###.##"; LC
TAX = C * .05: TAX = INT(TAX * 100 + .5) / 100
PRINT USING "5% TAX:      ###.##"; TAX
TOT = LC + C + TAX
PRINT USING "TOTAL:      ###.##"; INT(TOT * 100 + .5) / 100
```

```

DATA S193,10 INCH SPROCKET,13.95
DATA S867,30 INCH CHAIN,27.50
DATA F234,BLITZ MAG FRAME,119.00
DATA S445,COMPUTCYCLE COMPUTER,33.95
DATA C492,JET BRAKE SET,29.98
DATA J273,27 INCH WHEEL,32.00
DATA T100,27x1 INCH TIRE TUBE,12.50

```

'1.8

' This program will display labels alphabetically.

```

,
INPUT "Enter # of lines on label:"; H
S = 1: READ A$(S), B$(S)
WHILE A$(S) <> "*END*"
  L = LEN(A$(S)): I = 1
  WHILE MID$(A$(S), I, 1) <> " ": I = I + 1: WEND
  R$ = RIGHT$(A$(S), L - I): L$ = LEFT$(A$(S), I)
  C$(S) = R$ + ", " + L$
  S = S + 1
  READ A$(S), B$(S)
WEND
S = S - 1
FOR I = 1 TO S - 1
  FOR J = I + 1 TO S
    IF C$(I) > C$(J) THEN
      X$ = C$(I): C$(I) = C$(J): C$(J) = X$
      X$ = B$(I): B$(I) = B$(J): B$(J) = X$
    END IF
  NEXT J
NEXT I
FOR I = 1 TO S
  PRINT : PRINT C$(I): PRINT B$(I)
  FOR J = 1 TO H - 3: PRINT : NEXT J
NEXT I
DATA LISA SPINXS,987-6543
DATA BOB SIMON,123-4455
DATA BILL SIMON,123-4567
DATA HARRY TROUTMAN,876-2174
DATA HARRY PARKER,222-3333
DATA *END*,0

```

```
'1.9
' This program will guess secret letter in 5x5 matrix.
,
CLS : DIM A(25): S = 11
RANDOMIZE TIMER
FOR I = 1 TO 5
  FOR J = 1 TO 5
    DO: X = INT(RND(3) * 25): LOOP UNTIL A(X) = 0
    B$(I, J) = CHR$(X + 65): LOCATE I, 13 + J * 2: PRINT B$(I, J)
    A(X) = 1
  NEXT J
NEXT I
LOCATE 2, 30: PRINT "SCORE="; S
I = 0
WHILE A$ <> "Y"
  I = I + 1
  LOCATE 10, 10: PRINT "IS THE LETTER IN ROW"; I; : INPUT A$
  LOCATE 2, 30: S = S - 1: PRINT "SCORE="; S: LOCATE 10, 10
WEND: A$ = "": J = 0
WHILE A$ <> "Y"
  J = J + 1
  LOCATE 12, 10: PRINT "IS THE LETTER IN COL"; J; : INPUT A$
  LOCATE 2, 30: S = S - 1: PRINT "SCORE="; S
WEND
IF S > 0 THEN LOCATE 14, 1: PRINT "YOUR LETTER IS "; B$(I, J)
```

```

'1.10
' This program will display squares relative to cursor and #.
' Key movements: I-up, J-left, K-right, M-down
'
CLS : R = 5: C = 5
WHILE VAL(A$) = 0
  LOCATE R, C: PRINT "#": A$ = ""
  WHILE A$ = "": A$ = INKEY$: WEND
  IF VAL(A$) = 0 THEN
    LOCATE R, C: PRINT " "
    IF A$ = "I" THEN R = R - 1
    IF A$ = "M" THEN R = R + 1
    IF A$ = "J" THEN C = C - 1
    IF A$ = "K" THEN C = C + 1
  END IF
WEND
X = VAL(A$)
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
  LOCATE 21, 1: PRINT "OFF THE SCREEN": END
END IF
IF C + 9 * B > 80 OR C + 9 * B < 1 THEN
  LOCATE 21, 1: PRINT "OFF THE SCREEN": END
END IF
LOCATE R + 1 * A, C + 8 * B: PRINT "*****"
LOCATE R + 2 * A, C + 8 * B: PRINT "*      *"
LOCATE R + 3 * A, C + 8 * B: PRINT USING "*      #      *"; X
LOCATE R + 4 * A, C + 8 * B: PRINT "*      *"
LOCATE R + 5 * A, C + 8 * B: PRINT "*****"

```

'2.1

' This program will outline screen with random letters.

,

RANDOMIZE TIMER

DO

CLS

FOR I = 1 TO 11

X = INT(RND * 26): A\$ = CHR\$(65 + X)

LOCATE I, I

FOR J = I TO 80 - I: PRINT A\$; : NEXT J

FOR J = I + 1 TO 23 - I

LOCATE J, I: PRINT A\$: LOCATE J, 80 - I: PRINT A\$

NEXT J

LOCATE 23 - I, I

FOR J = I TO 80 - I: PRINT A\$; : NEXT J

B\$ = "": WHILE B\$ = "": B\$ = INKEY\$: WEND

NEXT I

LOOP UNTIL B\$ = CHR\$(27)

'2.2

' This program will print the longest sequence of letters.

,

INPUT "Enter N: "; N: DIM A\$(N)

FOR I = 1 TO N: INPUT "Enter letter: "; A\$(I): NEXT I

I = N: FOUND = 0

WHILE (I >= 2) AND NOT FOUND

FOR J = 1 TO N - I + 1

ONE = -1

FOR K = 0 TO I - 2

IF A\$(J + K) >= A\$(J + K + 1) THEN ONE = 0

NEXT K

IF ONE THEN

FOR K = 0 TO I - 1: PRINT A\$(J + K); " "; : NEXT K

PRINT : FOUND = -1

END IF

NEXT J

I = I - 1

WEND

'2.3

' This program will change the margins of a given text.

```

'
INPUT "Enter text: "; A$: A$ = A$ + " ": L = LEN(A$)
LW = 5: PRINT SPACE$(10);
FOR I = 1 TO L
  C$ = MID$(A$, I, 1)
  IF C$ <> " " THEN
    W$ = W$ + C$
  ELSE
    LL = LEN(W$)
    IF LW + LL > 30 THEN PRINT : PRINT SPACE$(5); : LW = 0
    IF LL > 0 THEN PRINT W$; " "; : LW = LW + LL + 1: W$ = ""
    IF LL = 0 AND LW > 0 THEN PRINT " "; : LW = LW + 1
  END IF
NEXT I

```

'2.4

' This program will print word with consonants alphabeitized.

```

'
INPUT "Enter word: "; A$: L = LEN(A$): V$ = "AEIOU"
DIM V$(L), C$(L), A(L)
FOR I = 1 TO L
  B$ = MID$(A$, I, 1): J = 1
  WHILE (J < 5) AND (MID$(V$, J, 1) <> B$): J = J + 1: WEND
  IF MID$(V$, J, 1) <> B$ THEN
    C = C + 1: C$(C) = B$
  ELSE
    V = V + 1: A(I) = 1: V$(V) = B$
  END IF
NEXT I
FOR I = 1 TO V - 1
  FOR J = I + 1 TO V
    IF V$(I) > V$(J) THEN X$ = V$(I): V$(I) = V$(J): V$(J) = X$
  NEXT J
NEXT I
FOR I = 1 TO C - 1
  FOR J = I + 1 TO C
    IF C$(I) > C$(J) THEN X$ = C$(I): C$(I) = C$(J): C$(J) = X$
  NEXT J
NEXT I
FOR I = 1 TO L
  IF A(I) = 1 THEN VV = VV + 1: PRINT V$(VV);
  IF A(I) = 0 THEN CC = CC + 1: PRINT C$(CC);
NEXT I

```

'2.5

' This program will print common letters and line up words.
,

```
INPUT "Enter N: "; N
FOR I = 1 TO N: INPUT "Enter word: "; A$(I): NEXT I
FOR I = 1 TO 26
  X$ = CHR$(64 + I): CO = -1: J = 1
  WHILE (J <= N) AND CO
    CO = INSTR(A$(J), X$)
    J = J + 1
  WEND
  IF CO THEN PRINT X$; " "; : FOUND = -1
NEXT I
IF NOT FOUND THEN PRINT "NO COMMON LETTERS": END
PRINT : INPUT "Choose letter: "; A$
FOR I = 1 TO N
  J = 1: WHILE MID$(A$(I), J, 1) <> A$: J = J + 1: WEND
  PRINT SPACE$(10 - J); A$(I)
NEXT I
```

```

'2.6
' This program will keep score for a double dual race.
,
CLS : DIM IN$(21)
FOR I = 1 TO 21
  PRINT "Place "; I; ":"; : INPUT IN$(I)
  IF I > 1 THEN
    J = 1
    WHILE J <= TN AND INIT$(J) <> IN$(I): J = J + 1: WEND
  END IF
  IF (INIT$(J) <> IN$(I)) OR (I = 1) THEN
    TN = TN + 1: INIT$(TN) = IN$(I)
  END IF
NEXT I
' Assert TEAM$(1, 2, 3) = 3 unique team INITIALS
FOR I = 1 TO 2
  FOR J = I + 1 TO 3
    PL = 0: T1 = 0: T2 = 0: T1PL = 0: T2PL = 0
    FOR K = 1 TO 21
      IF IN$(K) = INIT$(I) THEN
        PL = PL + 1: T1 = T1 + PL: T1PL = T1PL + 1
        TEAM1(T1PL) = PL
      END IF
      IF IN$(K) = INIT$(J) THEN
        PL = PL + 1: T2 = T2 + PL: T2PL = T2PL + 1
        TEAM2(T2PL) = PL
      END IF
    NEXT K
    T1 = T1 - TEAM1(6) - TEAM1(7)
    T2 = T2 - TEAM2(6) - TEAM2(7)
    PRINT "TEAM "; INIT$(I); ":"; T1; " POINTS"
    PRINT "TEAM "; INIT$(J); ":"; T2; " POINTS"
    IF (T1 < T2) OR (T1 = T2 AND TEAM1(6) < TEAM2(6)) THEN
      PRINT "TEAM "; INIT$(I);
    ELSE
      PRINT "TEAM "; INIT$(J);
    END IF
    PRINT " WINS!": PRINT
  NEXT J
NEXT I

```

'2.7

' This program will allow manipulation of 3x3 array of data.

```

DATA 10.11, 20.22, 30.33
DATA 11.1, 22.2, 33.3
DATA 10, 20, 30
FOR I = 1 TO 3: FOR J = 1 TO 3: READ A(I, J): NEXT J, I
WHILE C$ <> "C"
  CLS
  PRINT "A. EDIT OR CHANGE A VALUE"
  PRINT "B. DISPLAY THE RESULTS"
  PRINT "C. QUIT."
  INPUT "Enter option: "; C$
  IF C$ = "A" THEN
    INPUT "Enter row, col: "; ROW, COL
    INPUT "Enter number: "; A(ROW, COL)
  ELSEIF C$ = "B" THEN
    FOR I = 1 TO 3: A(I, 4) = 0: NEXT I
    FOR J = 1 TO 3: A(4, J) = 0: NEXT J: TOT = 0
    FOR I = 1 TO 3
      FOR J = 1 TO 3
        PRINT USING "###.## "; A(I, J); : TOT = TOT + A(I, J)
        A(4, J) = A(4, J) + A(I, J): A(I, 4) = A(I, 4) + A(I, J)
      NEXT J
      PRINT USING "###.##"; A(I, 4)
    NEXT I
    FOR J = 1 TO 3: PRINT USING "###.## "; A(4, J); : NEXT J
    PRINT USING "###.##"; TOT
  END IF
  IF C$ <> "C" THEN
    PRINT : PRINT "Press any key: ";
    A$ = "": WHILE A$ = "": A$ = INKEY$: WEND
  END IF
WEND

```

'2.8

' This program will print all combinations of 4 digits.

```

FOR A = 1 TO 8
  FOR B = A + 1 TO 9
    P = A * B
    IF P >= 10 THEN
      P$ = MID$(STR$(P), 2)
      C = VAL(MID$(P$, 1, 1)): D = VAL(MID$(P$, 2, 1))
      IF NOT (A = C OR A = D OR B = C OR B = D) THEN
        PRINT A; B; C; D, A; "X"; B; "= "; P$: S = S + 1
      END IF
    END IF
  NEXT B
NEXT A
PRINT " TOTAL ="; S

```

```
'2.9
' This program will select words, given a string w/wildcard.
'
INPUT "Enter N: "; N: DIM A$(N)
FOR I = 1 TO N: INPUT "Enter word: "; A$(I): NEXT I
DO
  INPUT "Enter string: "; A$: L = LEN(A$): W = 0
  I = INSTR(A$, "*"): IF I = 0 THEN END
  ' Asterisk is position I
  L$ = LEFT$(A$, I - 1): R$ = RIGHT$(A$, L - I)
  FOR J = 1 TO N
    IF LEFT$(A$(J), I - 1) = L$ THEN
      IF RIGHT$(A$(J), L - I) = R$ THEN
        PRINT A$(J): W = 1
      END IF
    END IF
  NEXT J
  IF W = 0 THEN PRINT "NO WORDS FOUND"
  PRINT
LOOP UNTIL I = 0
```

```
'2.10
' This program will maintain air conditioning in 3 rooms.
'
INPUT "Enter last 5-minutes: "; LM
CLS : OF = 72: CO = 65: DR = 79
PRINT " OF CO DS OFFICE COMP. DRY. MIN:SEC"
DO
  IF ((M MOD 5 = 0) AND S = 0) OR CH = 1 THEN
    PRINT O; C; D; " ";
    PRINT USING "##.#"; OF; : PRINT " ";
    PRINT USING "##.#"; CO; : PRINT " ";
    PRINT USING "##.#"; DR; : PRINT " ";
    PRINT USING "###: "; M;
    IF S > 0 THEN PRINT USING "##"; S ELSE PRINT "00"
    CH = 0
  END IF
  S = S + 15: IF S = 60 THEN M = M + 1: S = 0
  OF = OF + .1 - OFAIR
  CO = CO + .2 - COAIR
  DR = DR + .025 - DRAIR
  IF OF > 78 AND O = 0 THEN O = 1: CH = 1
  IF CO > 70 AND C = 0 THEN C = 1: CH = 1
  IF DR > 85 AND D = 0 THEN D = 1: CH = 1
  IF OF < 72 AND O = 1 THEN O = 0: CH = 1
  IF CO < 65 AND C = 1 THEN C = 0: CH = 1
  IF DR < 75 AND D = 1 THEN D = 0: CH = 1
  AIR = (O + C + D) * 2
  IF AIR = 0 THEN
    OFAIR = 0: COAIR = 0: DRAIR = 0
  ELSE
    OFAIR = O / AIR: COAIR = C / AIR: DRAIR = D / AIR
  END IF
LOOP UNTIL (M = LM) AND (S > 0)
```

```
'3.1
' This program will display the sides of a die.
'
' 6 ways to represent die (each with different top)
' DATA Top, Front, Right, Back, Left, (Bottom derived from top)
DATA 1, 5, 4, 2, 3
DATA 6, 5, 3, 2, 4
DATA 5, 1, 3, 6, 4
DATA 2, 1, 4, 6, 3
DATA 3, 5, 1, 2, 6
DATA 4, 5, 6, 2, 1
INPUT "Enter top, front: "; T, F
' Determine which data set to use (based on top #)
WHILE A <> T
  READ A
  FOR J = 1 TO 4: READ B(J): NEXT J
WEND
' Rotate sides till a side matches the front #
J = 1
WHILE B(J) <> F:
  J = J + 1
WEND
IF J = 4 THEN J = 0
R = J + 1
'Generate rest of sides, sum of opposites sides = 7
PRINT USING "TOP=# FRONT=# RIGHT=#"; T; F; B(R)
PRINT USING "BACK=# LEFT=# BOTTOM=#"; 7 - F; 7 - B(R); 7 - T
```

```
'3.2
' This program will factor a quadratic equation.
,
INPUT "Enter A, B, C: ", A, B, C
IF A < 0 THEN A = -A: B = -B: C = -C
IF A > 1 THEN
  FOR I = A TO 2 STEP -1
    IF (A MOD I = 0) AND (B MOD I = 0) AND (C MOD I = 0) THEN
      A = A / I: B = B / I: C = C / I
      PRINT LTRIM$(STR$(I));
    END IF
  NEXT I
END IF
S = B * B - 4 * A * C
IF S < 0 THEN PRINT "CANNOT BE FACTORED": END
H = INT(SQR(S) + .1): E = 2 * A
R(1) = -B + H: R(2) = -B - H
FOR K = 1 TO 2
  D = E: N = R(K)
  I = D
  WHILE (I > 0) AND ((N MOD I <> 0) OR (D MOD I <> 0))
    I = I - 1
  WEND
  N = N / I: D = D / I
  PRINT "("; : IF D > 1 THEN PRINT LTRIM$(STR$(D));
  PRINT "X";
  IF N < 0 THEN PRINT "+"; LTRIM$(STR$((-N)); ")";
  IF N > 0 THEN PRINT "-"; LTRIM$(STR$(N)); ")";
NEXT K
```

'3.3

' This program will simulate a calculator.

,

```
INPUT "Enter expression: "; A$: L = LEN(A$)
FOR I = 1 TO L
  B$ = MID$(A$, I, 1)
  IF ASC(B$) >= ASC("0") THEN
    C$ = C$ + B$
  ELSE
    J = J + 1: A(J) = VAL(C$): C$ = ""
    B(J) = INSTR("+-*/", B$)
  END IF
NEXT I
J = J + 1: A(J) = VAL(C$): K = 1
FOR I = 1 TO J - 1
  IF B(I) < 3 THEN
    B(K) = B(I): K = K + 1: A(K) = A(I + 1)
  ELSE
    IF B(I) = 3 THEN
      A(K) = A(K) * A(I + 1)
    ELSE
      A(K) = A(K) / A(I + 1)
    END IF
  END IF
NEXT I
S = A(1)
IF K > 1 THEN
  FOR I = 1 TO K - 1
    IF B(I) = 2 THEN S = S - A(I + 1) ELSE S = S + A(I + 1)
  NEXT I
END IF
PRINT USING "###.###"; S
```

'3.4

' This program will compute all digits of N factorial.

,

```
INPUT "Enter N: "; N: DIM A(3 * N)
D = 1: A(1) = 1
FOR I = 1 TO N
  FOR J = 1 TO D
    A(J) = A(J) * I + C: C = INT(A(J) / 10)
    A(J) = A(J) - 10 * C
  NEXT J
  WHILE C > 0
    CC = INT(C / 10): D = D + 1: A(D) = C - 10 * CC: C = CC
  WEND
NEXT I
FOR I = D TO 1 STEP -1: PRINT MID$(STR$(A(I)), 2); : NEXT I
```

```

'3.5
' This program will sum and subtract 2 big decimals.
,
DIM A(30), B(30), C(30), D(30): CLS
INPUT "Enter #1: "; A$: INPUT "Enter #2: "; B$
A = LEN(A$): B = LEN(B$)
FOR I = A TO 1 STEP -1
  IF MID$(A$, I, 1) = "." THEN
    X = I
  ELSE
    S = S + 1: A(S) = VAL(MID$(A$, I, 1))
  END IF
NEXT I: S = 0
FOR I = B TO 1 STEP -1
  IF MID$(B$, I, 1) = "." THEN
    Y = I
  ELSE
    S = S + 1: B(S) = VAL(MID$(B$, I, 1))
  END IF
NEXT I
' Align decimal point
G = A - X: H = B - Y
IF G > H THEN L = G ELSE L = H
Z = G - H
IF Z > 0 THEN
' Second # is smaller, so place leading 0s and align decimal
  FOR I = B - 1 TO 1 STEP -1
    B(I + Z) = B(I): B(I) = 0
  NEXT I
  B = B + Z
ELSEIF Z < 0 THEN
' First # is smaller, so place leading 0s and align decimal
  FOR I = A - 1 TO 1 STEP -1
    A(I - Z) = A(I): A(I) = 0
  NEXT I
  A = A - Z
END IF
IF A > B THEN Y = A - 1 ELSE Y = B - 1
' Add and subtract
FOR I = 1 TO Y
  C(I) = A(I) + B(I) + C: C = INT(C(I) / 10)
  C(I) = C(I) - C * 10
  D(I) = A(I) - B(I) - D
  IF D(I) < 0 THEN D = 1 ELSE D = 0
  D(I) = D(I) + D * 10
NEXT I
PRINT "SUM = "; : IF C > 0 THEN PRINT LTRIM$(STR$(C));
FOR I = Y TO 1 STEP -1
  IF I = L THEN PRINT ".";
  PRINT LTRIM$(STR$(C(I)));
NEXT I
PRINT : PRINT "DIFFERENCE = ";
FOR I = Y TO 1 STEP -1
  IF I = L THEN PRINT ".";
  PRINT LTRIM$(STR$(D(I)));

```

NEXT I

'3.6

' This program will control the movements of a snake.

,

CLS : DIM A(25, 81)

V = 12: H = 8: LOCATE V, H

FOR I = 8 TO 32

PRINT "*"; : A(V, I) = 1

A\$ = A\$ + "12": B\$ = B\$ + RIGHT\$(STR\$(I), 2)

NEXT I

WHILE D\$ = "": D\$ = INKEY\$: WEND: C\$ = D\$

DO UNTIL C\$ = CHR\$(27)

FOR I = 1 TO 100

D\$ = INKEY\$: IF D\$ <> "" THEN C\$ = D\$

NEXT I

IF C\$ = "I" THEN V = V - 1

IF C\$ = "M" THEN V = V + 1

IF C\$ = "J" THEN H = H - 1

IF C\$ = "K" THEN H = H + 1

IF A(V, H) OR V = 0 OR V = 25 OR H = 0 OR H = 81 THEN END

A(V, H) = 1: LOCATE V, H: PRINT "*"

X = VAL(RIGHT\$(A\$, 2)): Y = VAL(RIGHT\$(B\$, 2))

LOCATE X, Y: PRINT " "

A(X, Y) = 0

A\$ = LEFT\$(A\$, 24 * 2): B\$ = LEFT\$(B\$, 24 * 2)

A\$ = RIGHT\$(STR\$(V), 2) + A\$

B\$ = RIGHT\$(STR\$(H), 2) + B\$

LOOP

```

'3.7
' This program will print 3 permutations of a word.
'
INPUT "Enter word: "; A$: INPUT "Enter K: "; KK: L = LEN(A$)
FOR I = 1 TO L: A$(I) = MID$(A$, I, 1): NEXT I
' Alphabetize letters
FOR I = 1 TO L - 1
  FOR J = I + 1 TO L
    IF A$(I) > A$(J) THEN X$ = A$(I): A$(I) = A$(J): A$(J) = X$
  NEXT J
NEXT I
' Produce factorials F(I) = (I-1)!
FOR I = 1 TO L
  F = 1
  FOR J = 1 TO I - 1: F = F * J: NEXT J
  F(I) = F
NEXT I
FOR T = 1 TO 3
  K = KK * T - 1
  ' Generate Kth permutation
  FOR I = L TO 1 STEP -1
    X = INT(K / F(I))
    FOR J = 1 TO L
      IF A(J) = 0 THEN
        S = S + 1: IF S > X THEN A(J) = 1: PRINT A$(J); : J = L
      END IF
    NEXT J
    S = 0: K = K - F(I) * X
  NEXT I
  FOR I = 1 TO L: A(I) = 0: NEXT I
  PRINT " ";
NEXT T

```

'3.8

' This program will display N pennies on board.

```

'
INPUT "Enter N: "; N: DIM A(N): CLS
PRINT "TOTAL ="; N
IF N = 8 THEN SP = 1 ' 8 and 14 are special cases
J = N MOD 2: J = 2 - J: S = J
IF N = 14 THEN S = J + 2
PRINT " ";
FOR I = 1 TO N: PRINT USING "##"; I MOD 10; : NEXT I
PRINT
FOR I = 1 TO N: PRINT USING "#"; I MOD 10: NEXT I
FOR I = 1 TO N
  A(I) = S
  IF N = 14 AND I = 14 THEN S = 2: A(I) = S
  LOCATE 2 + I, 2 * S + 1: PRINT "*"
  S = S + 2 + SP
  IF S > N THEN
    IF SP THEN S = S - N ELSE S = (N MOD 2) + 1
  END IF
NEXT I
FOR I = 1 TO N
  LOCATE I + 2, 2 * N + 4: PRINT "(";
  PRINT LTRIM$(STR$(I)); ", "; LTRIM$(STR$(A(I))); ") ";
  PRINT "SUM ="; I + A(I)
NEXT I

```

'3.9

' This program will determine # of moves made to a stack.

```

'
' 1 block - 1 move (obvious)
' 2 blocks- 3 moves (Move 1 stack, move #2, move 1 stack)
' 3 blocks- 7 moves (Move 2 stack, move #3, move 2 stack on #3)
' (3 moves + 1 move + 3 moves)
' 4 blocks- 15 moves (Move 3 stack, move #4, move 3 stack on #4)
' (7 moves + 1 move + 7 moves)
' N blocks- 2^N - 1 moves
DIM A(16)
INPUT "Enter N: "; N: A(1) = 1
FOR I = 2 TO N: A(I) = A(I - 1) * 2 + 1: NEXT I
PRINT A(N)

```

```

'3.10
' This program will find set of #s P, Q, R (P = Q x R).
,
INPUT "Enter S:"; S
Q = S
DO
  DO
    Q = Q + 1: X1 = INT(Q / 10): Y1 = Q - X1 * 10
  LOOP UNTIL X1 <> Y1
  NU = INT(10000 / Q)
  FOR R = NU TO 999
    X2 = INT(R / 100): C = R - X2 * 100
    Y2 = INT(C / 10): Z2 = C - Y2 * 10
    IF X2 <> Y2 AND Y2 <> Z2 AND X2 <> Z2 THEN
      IF X1 <> X2 AND X1 <> Y2 AND X1 <> Z2 THEN
        IF Y1 <> X2 AND Y1 <> Y2 AND Y1 <> Z2 THEN
          A(X1) = 1: A(Y1) = 1: A(X2) = 1: A(Y2) = 1: A(Z2) = 1
          P$ = STR$(Q * R)
          FOR I = 2 TO 6
            X = VAL(MID$(P$, I, 1)): IF A(X) THEN DUPL = -1
          NEXT I
          FOR I = 2 TO 5
            FOR J = I + 1 TO 6
              IF MID$(P$, I, 1) = MID$(P$, J, 1) THEN DUPL = -1
            NEXT J
          NEXT I
          IF NOT DUPL THEN
            PRINT "P ="; P$;
            PRINT USING " Q = ## R = ###"; Q; R: END
          END IF
          FOR I = 0 TO 9: A(I) = 0: NEXT I: DUPL = 0
        END IF
      END IF
    END IF
  NEXT R
LOOP UNTIL Q > 99

```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '86
BASIC PROGRAM SOLUTIONS

'1.1

' This program will print "THIS IS THE EASIEST PROGRAM!".

,

CLS

A\$ = "THIS IS THE EASIEST PROGRAM!"

LOCATE 12, (80 - LEN(A\$)) / 2: PRINT A\$

'1.2

' This program will display the sum, difference, and product.

,

INPUT "Enter two numbers: "; A, B

PRINT "SUM = "; A + B

PRINT "DIFFERENCE = "; A - B

PRINT "PRODUCT = "; A * B

'1.3

' This program will sum $1 + (1/2)^2 + (1/3)^3 + (1/4)^4 + \dots$

' until the difference between it and the next term is within E.

,

INPUT "Enter test value E: "; E

I = 1

SUM = 1: LSUM = 0

WHILE (SUM - LSUM) >= E

I = I + 1

TRM = 1 / I: PROD = 1

FOR J = 1 TO I: PROD = PROD * TRM: NEXT J

LSUM = SUM

SUM = SUM + PROD

WEND

PRINT USING "#.#####"; LSUM

```
'1.4
' This program will print a check given name and amount.
'
```

```
CLS
INPUT "Enter first name: "; F$
INPUT "Enter middle name: "; M$
INPUT "Enter last name: "; L$
I$ = LEFT$(M$, 1)
INPUT "Enter amount: "; AMOUNT$
' Display border
LOCATE 6, 1
PRINT STRING$(39, "*")
FOR I = 1 TO 9
  LOCATE 6 + I, 1: PRINT "*"
  LOCATE 6 + I, 39: PRINT "*"
NEXT I
PRINT STRING$(39, "*")
'
LOCATE 8, 3: PRINT "BEN'S TOWING SERVICE"
LOCATE 9, 3: PRINT "4563 WRECKER AVENUE"
LOCATE 10, 3: PRINT "WAVERLY, ARKANSAS 45632"
LOCATE 12, 4: PRINT "PAY TO THE ORDER OF ";
PRINT F$; " "; I$; ". "; L$
LOCATE 14, 4: PRINT "THE SUM OF $"; AMOUNT$
LOCATE 22, 1
```

```
'1.5
' This program will determine which prisoners may be released.
'
```

```
DIM CELL(100)
FOR I = 1 TO 100: CELL(I) = 1: NEXT I      'Cells initially open
FOR I = 2 TO 100
  J = 1
  WHILE J <= 100
    CELL(J) = 1 - CELL(J): J = J + I
  WEND
NEXT I
FOR I = 1 TO 100
  IF CELL(I) = 1 THEN PRINT "CELL"; I
NEXT I
```

'1.6

' This program will determine how much money accumulates.
' Double precision variables (#) are needed.

```
,  
INPUT "Enter monthly investment: "; MONTH#  
INPUT "Enter end of year deposit: "; DEP#  
INPUT "Enter annual rate of interest: "; RATE#  
PRINT  
RATE# = RATE# / (12 * 100) 'Rate per month in yr in percent  
FOR YEAR = 1 TO 20  
  FOR J = 1 TO 12  
    SUM# = SUM# + MONTH#  
    SUM# = SUM# + RATE# * SUM#  
  NEXT J  
  SUM# = SUM# + DEP#  
NEXT YEAR  
SUM# = INT(SUM# * 100 + .5) / 100  
PRINT "AMOUNT AT END OF YEAR 20 IS $"; LTRIM$(STR$(SUM#))
```

'1.7

' This program will drop g in words ending with ing or ings.

```
,  
INPUT "Enter sentence: "; S$  
S$ = S$ + " "  
L = LEN(S$): W$ = ""  
FOR I = 1 TO L  
  CH$ = MID$(S$, I, 1)  
  IF CH$ <> " " THEN  
    W$ = W$ + CH$  
  ELSE  
    LENW = LEN(W$)  
    IF LENW >= 4 THEN  
      EN1$ = MID$(W$, LENW - 2, 3)  
      EN2$ = MID$(W$, LENW - 3, 4)  
      IF EN1$ = "ING" THEN W$ = MID$(W$, 1, LENW - 1)  
      IF EN2$ = "INGS" THEN W$ = MID$(W$, 1, LENW - 2) + "S"  
    END IF  
    PRINT W$; " ";  
    W$ = ""  
  END IF  
NEXT I
```

```
'1.8
' This program simulates the population growth of rabbits.
,
INPUT "Enter initial population: "; INIT
INPUT "Enter point of over population: "; OP
PRINT
POP = INIT
DIEING = (POP >= OP)
FOR MONTH = 1 TO 23
  IF DIEING THEN
    IF POP < 2 / 3 * INIT THEN
      POP = POP + POP * .2: DIEING = 0
    ELSE
      POP = POP - POP * .15
    END IF
  ELSE
    IF POP >= OP THEN
      DIEING = -1: INIT = INT(POP)
      POP = POP - POP * .15
    ELSE
      POP = POP + POP * .2
    END IF
  END IF
  PRINT "POPULATION FOR MONTH"; MONTH; "IS"; INT(POP + .5)
NEXT MONTH
```

```
'1.9
' This program doubles every e that appears as a single e.
,
INPUT "Enter sentence: "; SENT$
FOR I = 1 TO LEN(SENT$)
  CH$ = MID$(SENT$, I, 1)
  NCH$ = MID$(SENT$, I + 1, 1)
  IF CH$ = "E" AND LCH$ <> "E" AND NCH$ <> "E" THEN PRINT "E";
  PRINT CH$;
  LCH$ = CH$
NEXT I
IF NCH$ = "E" AND LCH$ <> "E" THEN PRINT "E";
PRINT NCH$
```

```
'1.10
' This program will display common elements of two lists.
'
DIM A(12), B(12), C(12)
FOR I = 1 TO 12
    PRINT "Enter"; I; "of 12: "; : INPUT A(I)
NEXT I
FOR I = 1 TO 11
    PRINT "Enter"; I; "of 11: "; : INPUT B(I)
NEXT I
'
FOR I = 1 TO 12
    FOR J = 1 TO 11
        IF A(I) = B(J) THEN C(I) = 1
    NEXT J
NEXT I
FOR I = 1 TO 12
    FOR J = I + 1 TO 12
        IF A(I) = A(J) AND C(J) > 0 THEN C(J) = C(J) + 1
    NEXT J
NEXT I
FOR I = 1 TO 12
    IF C(I) = 1 THEN PRINT A(I); " ";
NEXT I
```

```
'2.1
' This program will right justify sentence within 65 columns.
,
COL = 65
INPUT "Enter sentence: "; SENT$
SENT$ = SENT$ + " ": L = LEN(SENT$)
I = 1: WN = 1: WORD$(WN) = "": TOTCH = 0
WHILE I <= L
  CH$ = MID$(SENT$, I, 1)
  IF CH$ <> " " THEN
    WORD$(WN) = WORD$(WN) + CH$
  ELSE
    IF WORD$(WN) <> "" THEN
      TOTCH = TOTCH + LEN(WORD$(WN))
      WN = WN + 1: WORD$(WN) = ""
    END IF
  END IF
  I = I + 1
WEND
WN = WN - 1
,
SPACE = INT((COL - TOTCH) / (WN - 1))
EXTRA = (COL - TOTCH) - (SPACE * (WN - 1))
FOR I = 1 TO WN
  IF I <= EXTRA THEN EX = 1 ELSE EX = 0
  PRINT WORD$(I); SPACE$(SPACE + EX);
NEXT I
```

```
'2.2
' This program will produce a repeating pattern of XXX ---.
,
INPUT "Enter total number of X's and -'s: "; TOTALXD
INPUT "Enter number of X's: "; NUMX
INPUT "Enter number of rows: "; ROWS
X1$ = "": X2$ = "": D1$ = "": D2$ = ""
FOR I = 1 TO NUMX
  X1$ = X1$ + "X"
  D2$ = D2$ + "-"
NEXT I
FOR I = 1 TO TOTALXD - NUMX
  X2$ = X2$ + "X"
  D1$ = D1$ + "-"
NEXT I
FOR ROW = 1 TO ROWS
  IF ROW - INT(ROW / 2) * 2 = 1 THEN
    FOR I = 1 TO 4: PRINT X1$; D1$; : NEXT I
  ELSE
    FOR I = 1 TO 4: PRINT D2$; X2$; : NEXT I
  END IF
  PRINT
NEXT ROW
```

'2.3

' This program will code or decode a message.

,

```
ST1$ = "ZXCVBNMASDFGHJKLQWERTYUIOP "
```

```
ST2$ = "ABCDEFGHJKLMNOPQRSTUVWXYZ "
```

```
WHILE OP < 3
```

```
  PRINT
```

```
  PRINT "1) ENCODE"
```

```
  PRINT "2) DECODE"
```

```
  PRINT "3) END"
```

```
  INPUT "Choose: "; OP
```

```
  IF OP = 3 THEN END
```

```
  INPUT "Enter message: "; MESSAGE$
```

```
  FOR I = 1 TO LEN(MESSAGE$)
```

```
    CH$ = MID$(MESSAGE$, I, 1)
```

```
    IF CH$ <> " " THEN
```

```
      IF OP = 1 THEN
```

```
        CH$ = MID$(ST1$, ASC(CH$) - 64, 1)
```

```
      ELSE
```

```
        J = INSTR(ST1$, CH$)
```

```
        CH$ = MID$(ST2$, J, 1)
```

```
      END IF
```

```
    END IF
```

```
    PRINT CH$;
```

```
  NEXT I
```

```
  PRINT
```

```
WEND
```

'2.4

' This program finds the unique mode of a set of 15 numbers.

,

```
DIM A(15), C(15)
```

```
FOR I = 1 TO 15
```

```
  PRINT "Enter number"; I; ": "; : INPUT A(I)
```

```
NEXT I
```

```
MAX = 1
```

```
FOR I = 1 TO 14
```

```
  C(I) = 1
```

```
  FOR J = I + 1 TO 15
```

```
    IF A(I) = A(J) THEN
```

```
      C(I) = C(I) + 1
```

```
      IF C(I) > MAX THEN MAX = C(I)
```

```
    END IF
```

```
  NEXT J
```

```
NEXT I
```

```
MODEXIST = 0
```

```
FOR I = 1 TO 14
```

```
  IF C(I) = MAX THEN
```

```
    IF MODEXIST THEN PRINT "NO UNIQUE MODE": END
```

```
    MODE = A(I): MODEXIST = -1
```

```
  END IF
```

```
NEXT I
```

```
IF MODEXIST THEN PRINT "MODE IS"; MODE: END
```

```
PRINT "NO UNIQUE MODE"
```

```
'2.5
' This program simulates transactions to savings a account.
,
RATE = .07
INPUT "Enter original balance: "; BALANCE
WHILE OP < 4
  PRINT
  PRINT "1. MAKE A DEPOSIT"
  PRINT "2. MAKE A WITHDRAWAL"
  PRINT "3. CREDIT INTEREST"
  PRINT "4. END"
  INPUT "Enter option: "; OP
  PRINT
  IF OP = 1 THEN
    INPUT "Enter amount to deposit: "; DEP
    PRINT USING "BALANCE BEFORE TRANSACTION $####.##"; BALANCE
    BALANCE = BALANCE + DEP
    PRINT "MAKE A DEPOSIT"
  ELSEIF OP = 2 THEN
    INPUT "Enter amount to withdraw: "; WIT
    PRINT USING "BALANCE BEFORE TRANSACTION $####.##"; BALANCE
    BALANCE = BALANCE - WIT
    PRINT "MAKE A WITHDRAWAL"
  ELSEIF OP = 3 THEN
    PRINT USING "BALANCE BEFORE TRANSACTION $####.##"; BALANCE
    CREDIT = BALANCE * RATE / 12
    CREDIT = INT(CREDIT * 100 + .5) / 100
    PRINT USING "CREDIT INTEREST OF $##.##"; CREDIT
    BALANCE = BALANCE + CREDIT
  END IF
  IF OP < 4 THEN PRINT "NEW "; ELSE PRINT "FINAL ";
  PRINT USING "BALANCE $####.##"; BALANCE
WEND
```

'2.6

' This program will sum two positive big numbers.

'

DIM A(39), B(39), C(39)

INPUT "ENTER FIRST NUMBER: "; ST1\$

INPUT "ENTER SECOND NUMBER: "; ST2\$

L1 = LEN(ST1\$): L2 = LEN(ST2\$)

FOR I = 1 TO L1

 CH\$ = MID\$(ST1\$, L1 - I + 1, 1)

 A(I) = VAL(CH\$)

NEXT I

FOR I = 1 TO L2

 CH\$ = MID\$(ST2\$, L2 - I + 1, 1)

 B(I) = VAL(CH\$)

NEXT I

'

IF L1 > L2 THEN MAXL = L1 ELSE MAXL = L2

FOR I = 1 TO MAXL

 C(I) = A(I) + B(I) + CARRY

 IF C(I) > 9 THEN C(I) = C(I) - 10: CARRY = 1 ELSE CARRY = 0

NEXT I

IF CARRY = 1 THEN MAXL = MAXL + 1: C(MAXL) = 1

PRINT "SUM IS ";

FOR I = MAXL TO 1 STEP -1

 PRINT USING "#"; C(I);

NEXT I

```

'2.7
' This program will perform conversions.
,
DATA "INCHES", "FEET", "MILES", "OUNCES", "POUNDS", "GALLONS"
FOR I = 1 TO 6: READ DEC$(I): NEXT I
DATA 2.54, 0.3048, 1.6093, 28.35, 0.4536, 3.7854
FOR I = 1 TO 6: READ CON(I): NEXT I
DATA "CENTIMETERS", "METERS", "KILOMETERS", "GRAMS"
DATA "KILOGRAMS", "LITERS"
FOR I = 1 TO 6: READ MET$(I): NEXT I
,
WHILE OP <> 13
  PRINT
  FOR I = 1 TO 6
    PRINT I;
    IF I - INT(I / 2) * 2 = 1 THEN
      ST$ = MET$(INT((I + 1) / 2)) + " TO "
      ST$ = ST$ + DEC$(INT((I + 1) / 2))
      PRINT ST$; SPACE$(23 - LEN(ST$));
      PRINT USING "## "; I + 6;
      ST$ = MET$(INT((I + 7) / 2)) + " TO "
      ST$ = ST$ + DEC$(INT((I + 7) / 2))
    ELSE
      ST$ = DEC$(INT(I / 2)) + " TO "
      ST$ = ST$ + MET$(INT(I / 2))
      PRINT ST$; SPACE$(23 - LEN(ST$));
      PRINT USING "## "; I + 6;
      ST$ = DEC$(INT((I + 6) / 2)) + " TO "
      ST$ = ST$ + MET$(INT((I + 6) / 2))
    END IF
    PRINT ST$
  NEXT I
  PRINT SPACE$(26); "13 END"
  INPUT "Enter option: "; OP
  IF OP < 13 THEN
    IF OP - INT(OP / 2) * 2 = 1 THEN
      PRINT "Enter number of "; MET$(INT((OP + 1) / 2));
      INPUT ": "; X
      Y = X / CON(INT((OP + 1) / 2))
      PRINT USING "THIS IS EQUIVALENT TO ###.### "; Y;
      PRINT DEC$(INT((OP + 1) / 2))
    ELSE
      PRINT "Enter number of "; DEC$(INT(OP / 2));
      INPUT ": "; X
      Y = X * CON(INT(OP / 2))
      PRINT USING "THIS IS EQUIVALENT TO ###.### "; Y;
      PRINT MET$(INT(OP / 2))
    END IF
  END IF
END IF
WEND

```

```

'2.8
' This program will generate a mortgage amortization.
' Double precision variables are needed.
'
INPUT "Enter principal: "; PRINC#
INPUT "Enter % rate of interest: "; RATE#
INPUT "Enter term in years: "; YEARS
INPUT "Enter # of month in year for first payment: "; MONTH
RATE# = RATE# / (12 * 100): AMOUNT# = 1
FOR I = 1 TO YEARS * 12: AMOUNT# = AMOUNT# * (1 + RATE#): NEXT I
PAYMENT# = (RATE# * AMOUNT#) / (AMOUNT# - 1) * PRINC#
C = MONTH - 1: OLDP# = PRINC#
RATE# = RATE# * 12
PRINT "INTEREST          PRINCIPAL"
'
FOR I = 1 TO YEARS * 12
  MI# = OLDP# * RATE# / 12
  MP# = PAYMENT# - MI#
  OLDP# = OLDP# - MP#
  PRINT USING "$###.##"; MI#; : PRINT SPACES(10);
  PRINT USING "$#####.##"; OLDP#
  C = C + 1: YI# = YI# + MI#
  IF C - INT(C / 12) * 12 = 0 THEN
    PRINT
    PRINT USING "YEAR'S INTEREST  $#####.##"; YI#
    TI# = TI# + YI#: YI# = 0
    PRINT
    A$ = INPUT$(1)
  END IF
NEXT I
IF MONTH <> 1 THEN
  PRINT
  PRINT USING "YEAR'S INTEREST  $#####.##"; YI#
  TI# = TI# + YI#
END IF
PRINT USING "TOTAL INTEREST    $#####.##"; TI#
PRINT USING "MONTHLY PAYMENT   $#####.##"; PAYMENT#

```

'2.9

' This program calculates the value of sine(x) by a series.
' Double precision variables are needed.

```

INPUT "Enter N degrees: "; N
PI# = 3.1415926535#
IF N > 180 THEN X# = PI# * ((360 - N) / 180)
IF N <= 180 THEN X# = PI# * (N / 180)
POWER = -1
FOR I = 1 TO 6
  POWER = POWER + 2: FACT = 1
  FOR J = 1 TO POWER: FACT = FACT * J: NEXT J
  TRM# = 1
  FOR J = 1 TO POWER: TRM# = TRM# * X#: NEXT J
  TRM# = TRM# / FACT
  IF I - INT(I / 2) * 2 = 1 THEN
    SUM# = SUM# + TRM#
  ELSE
    SUM# = SUM# - TRM#
  END IF
NEXT I
IF N > 180 THEN SUM# = -1 * SUM#: X# = PI# * (N / 180)
PRINT "PARTIAL SUM ="; : IF SUM# < 0 THEN PRINT " ";
PRINT USING "##.#####"; SUM#
PRINT "ACTUAL SINE ="; : IF SIN(X#) < 0 THEN PRINT " ";
PRINT USING "##.#####"; SIN(X#)

```

'2.10

' This program will convert a Roman Numeral to Arabic form.

```

DATA M,1000, D,500, C,100, L,50, X,10, V,5, I,1
FOR I = 1 TO 7: READ RN$(I), RV(I): NEXT I
INPUT "Enter Roman Numeral: "; ROMNUM$
L = LEN(ROMNUM$): I = 1: ARABIC = 0
WHILE I < L
  FOR J = 1 TO 7
    IF MID$(ROMNUM$, I, 1) = RN$(J) THEN IND1 = J
    IF MID$(ROMNUM$, I + 1, 1) = RN$(J) THEN IND2 = J
  NEXT J
  IF IND1 <= IND2 THEN
    ARABIC = ARABIC + RV(IND1)
  ELSE
    ARABIC = ARABIC + RV(IND2) - RV(IND1): I = I + 1
  END IF
  I = I + 1
WEND
IF I = L THEN
  FOR J = 1 TO 7
    IF MID$(ROMNUM$, I, 1) = RN$(J) THEN IND1 = J
  NEXT J
  ARABIC = ARABIC + RV(IND1)
END IF
PRINT "ARABIC ="; ARABIC

```

'3.1

' This program produces montly calendars for the year 1986.

```

DATA JANUARY, FEBRUARY, MARCH, APRIL, MAY, JUNE, JULY
DATA AUGUST, SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER
DATA 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
DATA S, M, T, W, T, F, S
DIM MO$(12), DAYS(12)
FOR I = 1 TO 12: READ MO$(I): NEXT I
FOR I = 1 TO 12: READ DAYS(I): NEXT I
FOR I = 1 TO 7: READ D$(I): NEXT I
CLS : PRINT SPACE$(12); "1986": PRINT
FOR M = 1 TO 12
  IF M > 1 THEN CLS
  PRINT SPACE$(13 - INT(LEN(MO$(M)) / 2)); MO$(M): PRINT
  FOR I = 1 TO 7: PRINT " "; D$(I); " "; : NEXT I
  PRINT
  IF M = 1 THEN COL = 4
  IF COL > 1 THEN PRINT SPACE$((COL - 1) * 4);
  FOR DAY = 1 TO DAYS(M)
    PRINT USING "##"; DAY; : PRINT " ";
    IF COL < 7 THEN COL = COL + 1 ELSE COL = 1: PRINT
  NEXT DAY
  A$ = "": WHILE A$ = "": A$ = INKEY$: WEND
NEXT M

```

'3.2

' This program finds the root of a 5th degree polynomial
' of the form $Ax^5 + Bx^4 + Cx^3 + Dx^2 + Ex + F = 0$.

```

INPUT "Enter coefficients A,B,C,D,E,F: "; A, B, C, D, E, F
DEF FNY (Y) = C * Y ^ 3 + D * Y * Y + E * Y + F
DEF FNP (X) = A * X ^ 5 + B * X ^ 4 + FNY(X)
' This algorithm finds 1 and only 1 root (closest to x=0)
X1 = -1: X2 = 1
' Find sign change between X1 and X2
WHILE FNP(X1) * FNP(X2) > 0
  X1 = X1 - 1: X2 = X2 + 1
WEND
' Use binary search to find root
WHILE X2 - X1 > .000005
  X = (X1 + X2) / 2
  IF FNP(X) * FNP(X1) > 0 THEN X1 = X ELSE X2 = X
WEND
PRINT "ROOT = ";
IF X < 0 THEN PRINT "-"; : X = -X
PRINT USING "#.#####"; X

```

```
'3.3
' This program changes a number from one base to another.
,
D$ = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
INPUT "Enter base A: "; A
INPUT "Enter base B: "; B
INPUT "Enter original number: "; NUMST$
PRINT : PRINT NUMST$; " BASE"; A; "EQUALS ";
FOR I = 1 TO LEN(NUMST$)
    POW = INT(A ^ (LEN(NUMST$) - I) + .01)
    N = N + (INSTR(D$, MID$(NUMST$, I, 1)) - 1) * POW
NEXT I
POW = 1
WHILE POW <= N
    EX = EX + 1: POW = POW * B
WEND
EX = EX - 1
' Convert Num to Base B from Base 10
FOR I = EX TO 0 STEP -1
    POW = POW / B
    X = INT(N / POW + .01)
    PRINT MID$(D$, X + 1, 1);
    N = N - X * POW
NEXT I
PRINT " BASE"; B
```

```

'3.4
' This program will update customers account by SSN's.
,
DATA 234567890,"JOHN SMITH  "
DATA "1234 ANYWHERE LANE, EXIST, KANSAS 66754  ",345.78
DATA 564783219,"GAIL HUSTON  "
DATA "543 SOUTH THIRD, BIG TOWN, TEXAS 88642  ",2365.89
DATA 873421765,"TIM JONES  "
DATA "2387 PALM PLACE, NOME, ALASKA 77643  ",6754.76
DATA 543876543,"JILL RUPERTS"
DATA "4536 123RD STREET, TINY TOWN, MAINE 76765 ",45.18
DATA 345212342,"AL BROWN  "
DATA "PO BOX 234, TINSEL TOWN, CALIFORNIA 77654 ",3456.09
DATA 565656565,"KERMIT TEU  "
DATA "1234 LOST LANE, WIMPLE, WISCONSIN 66543  ",78.36
FOR I = 1 TO 6: READ SS$(I), N$(I), A$(I), B(I): NEXT I
INPUT "Enter SSN: "; SSN$
WHILE SSN$ <> "000000000"
  I = 1
  WHILE (SS$(I) <> SSN$) AND (I < 6): I = I + 1: WEND
  INPUT "Enter C for Charge or P for Payment: "; CH$
  INPUT "Enter amount of transaction: "; TRANS
  IF CH$ = "C" THEN B(I) = B(I) - TRANS
  IF CH$ = "P" THEN B(I) = B(I) + TRANS
  PRINT : PRINT USING "NEW BALANCE IS $####.##"; B(I)
  PRINT : INPUT "Enter SSN: "; SSN$
WEND
FOR I = 1 TO 5
  FOR J = I + 1 TO 6
    IF B(I) < B(J) THEN
      SWAP SS$(I), SS$(J)
      SWAP N$(I), N$(J)
      SWAP A(I), A(J)
      SWAP B(I), B(J)
    END IF
  NEXT J
NEXT I
PRINT
PRINT "SSN          NAME          ADDRESS"; SPACE$(13);
PRINT "BALANCE": PRINT
FOR I = 1 TO 6
  PR$ = SS$(I) + "  " + N$(I) + "  "
  L = LEN(PR$) - 1
  P1 = INSTR(A$(I), ",")
  P2 = INSTR(P1 + 1, A$(I), ",")
  PRINT PR$; LEFT$(A$(I), P1 - 1); SPACE$(21 - P1);
  PRINT USING "$####.##"; B(I)
  PRINT SPACE$(L); MID$(A$(I), P1 + 1, P2 - P1 - 1)
  PRINT SPACE$(L); MID$(A$(I), P2 + 1)
NEXT I

```

'3.5

' This program will print the product of 2 large decimals.

,

```
DIM A(30), B(30), PROD(50)
INPUT "Enter first number: "; ASTR$
INPUT "Enter second number: "; BSTR$
ADEC = INSTR(STR$, "."): BDEC = INSTR(BSTR$, ".")
ASTR$ = LEFT$(ASTR$, ADEC - 1) + RIGHT$(ASTR$, LEN(STR$) - ADEC)
BSTR$ = LEFT$(BSTR$, BDEC - 1) + RIGHT$(BSTR$, LEN(BSTR$) - BDEC)
LENA = LEN(STR$): LENB = LEN(BSTR$)
RDIGITS = LENA - ADEC + LENB - BDEC + 2
FOR I = LENA TO 1 STEP -1
    A(LENA - I + 1) = VAL(MID$(STR$, I, 1))
NEXT I
FOR I = LENB TO 1 STEP -1
    B(LENB - I + 1) = VAL(MID$(BSTR$, I, 1))
NEXT I
FOR I = 1 TO LENB
    CARRY = 0
    FOR J = 1 TO LENA
        S = I + J - 1
        PROD(S) = PROD(S) + B(I) * A(J) + CARRY
        CARRY = INT(PROD(S) / 10)
        PROD(S) = PROD(S) - CARRY * 10
    NEXT J
    IF CARRY > 0 THEN PROD(S + 1) = CARRY
NEXT I
PRINT "PRODUCT = ";
IF CARRY > 0 THEN S = S + 1
IF S <= RDIGITS THEN PRINT "0";
FOR I = S TO 1 STEP -1
    IF I = RDIGITS THEN PRINT ".";
    PRINT USING "#"; PROD(I);
NEXT I
```

```
'3.6
' This program will determine if a # can become palindrome.
,
DIM B(50), REV(50)
INPUT "Enter number: "; NUMST$
L = LEN(NUMST$)
FOR I = 1 TO L
  B(L - I + 1) = VAL(MID$(NUMST$, I, 1))
NEXT I
TRY = 0: PAL = 0
WHILE (TRY <= 23) AND (NOT PAL)
  PAL = -1
  FOR I = 1 TO INT(L / 2)
    IF B(I) <> B(L - I + 1) THEN PAL = 0
  NEXT I
' Add reverse of number to itself
IF NOT PAL THEN
  FOR I = 1 TO L: REV(I) = B(L - I + 1): NEXT I
  CARRY = 0
  FOR I = 1 TO L
    B(I) = B(I) + REV(I) + CARRY
    CARRY = INT(B(I) / 10)
    B(I) = B(I) - CARRY * 10
  NEXT I
  IF CARRY = 1 THEN L = L + 1: B(L) = 1
  TRY = TRY + 1
END IF
WEND
IF NOT PAL THEN PRINT "CANNOT GENERATE A PALINDROME": END
FOR I = L TO 1 STEP -1: PRINT USING "#"; B(I); : NEXT I
PRINT " IS A PALINDROME"
```

```
'3.7
' This program will solve an N x N system of equations.
'
INPUT "Enter N: "; N
FOR ROW = 1 TO N
  PRINT "Enter coefficients for row"; ROW
  FOR COL = 1 TO N
    PRINT USING "Co#"; COL; : PRINT ": ";
    INPUT C(ROW, COL)
  NEXT COL
  INPUT "Enter constant: "; C(ROW, N + 1)
NEXT ROW
'   Make main diagonals all 1s with 0s to the left
FOR ROW = 1 TO N
  DEN = C(ROW, ROW)
  FOR COL = ROW TO N + 1
    C(ROW, COL) = C(ROW, COL) / DEN
  NEXT COL
  FOR R = ROW + 1 TO N
    X = C(R, ROW)
    FOR COL = ROW TO N + 1
      C(R, COL) = C(R, COL) - X * C(ROW, COL)
    NEXT COL
  NEXT R
NEXT ROW
'   Make 0s on the right of 1s on main diagonal, not const
FOR ROW = N TO 1 STEP -1
  FOR R = ROW - 1 TO 1 STEP -1
    X = C(R, ROW)
    FOR COL = ROW TO N + 1
      C(R, COL) = C(R, COL) - X * C(ROW, COL)
    NEXT COL
  NEXT R
NEXT ROW
'   Display solution
PRINT "("; LTRIM$(STR$(INT(C(1, N + 1) + .1)));
FOR ROW = 2 TO N
  PRINT ", "; LTRIM$(STR$(INT(C(ROW, N + 1) + .1)));
NEXT ROW
PRINT ")"
```

```
'3.8
' This program prints Kth, 2*Kth, and 3*Kth permutations.
'
INPUT "Enter word: "; A$: INPUT "Enter K: "; KK: L = LEN(A$)
FOR I = 1 TO L: A$(I) = MID$(A$, I, 1): NEXT I
' Alphabetize letters
FOR I = 1 TO L - 1
  FOR J = I + 1 TO L
    IF A$(I) > A$(J) THEN X$ = A$(I): A$(I) = A$(J): A$(J) = X$
  NEXT J
NEXT I
' Produce factorials F(I) = (I-1)!
FOR I = 1 TO L
  F = 1
  FOR J = 1 TO I - 1: F = F * J: NEXT J
  F(I) = F
NEXT I
FOR T = 1 TO 3
  K = KK * T - 1
  ' Generate Kth permutation
  FOR I = L TO 1 STEP -1
    X = INT(K / F(I))
    FOR J = 1 TO L
      IF A(J) = 0 THEN
        S = S + 1: IF S > X THEN A(J) = 1: PRINT A$(J); : J = L
      END IF
    NEXT J
    S = 0: K = K - F(I) * X
  NEXT I
  FOR I = 1 TO L: A(I) = 0: NEXT I
  PRINT " ";
NEXT T
```

'3.9

```
' This program will solve cryptarithm puzzle ABB - CB = DEF.
' F = 0 since B-B = 0.  A=D+1 or A=D since CB is 2 digits,
' but A<>D.  D>B, otherwise D=A.  Since B<C, B<9, => E=10+B-C
'
FOR B = 1 TO 8
  FOR C = B + 1 TO 9
    FOR D = 1 TO 8
      F = 0: A = D + 1: E = 10 + B - C
      IF A = B OR A = C OR A = D OR A = E OR A = F THEN PASS = 1
      IF B = C OR B = D OR B = E OR B = F OR C = D THEN PASS = 1
      IF C = E OR C = F OR D = E OR D = F THEN PASS = 1
      IF PASS = 0 THEN
        TOT = TOT + 1
        PRINT A * 100 + B * 10 + B; "-"; C * 10 + B; "=";
        PRINT D * 100 + E * 10 + F; " NUMBER"; TOT
      ELSE
        PASS = 0
      END IF
    NEXT D
  NEXT C
NEXT B
PRINT : PRINT " TOTAL NUMBER OF SOLUTIONS ="; TOT
```

'3.10

```
' This program will find all 2-digit integers equal to the sum
' of integers in which each digit 0-9 is used exactly once.
'
FOR I = 0 TO 8
  ' Place digit I in front of 0 and sum the rest of the digits
  SUM = I * 10 + 0
  FOR J = 0 TO 9
    IF (I <> J) AND (J <> 0) THEN
      TRM = J: SUM = SUM + J
    END IF
  NEXT J
  IF SUM <= 99 THEN
    ' Display sum followed by example sum process
    PRINT SUM; "=";
    PRINT I * 10 + 0;
    FOR J = 0 TO 9
      IF (I <> J) AND (J <> 0) THEN
        TRM = J: PRINT "+"; J;
      END IF
    NEXT J
    PRINT
  END IF
NEXT I
```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '87
BASIC PROGRAM SOLUTIONS

'1.1

' This program will print out the sign of a given number.

,

```
INPUT "Enter number:"; N
IF N > 0 THEN PRINT "POSITIVE"
IF N < 0 THEN PRINT "NEGATIVE"
IF N = 0 THEN PRINT "ZERO"
```

'1.2

' This program will sum the numbers n, n+1, ... n+20.

,

```
INPUT "Enter n:"; N
FOR I = 0 TO 20
    SUM = SUM + N + I
NEXT I
PRINT "SUM ="; SUM
```

'1.3

' This program will print PROBLEM THREE diagonally.

,

```
CLS
A$ = "PROBLEM THREE"
L = LEN(A$)
ROW = (24 - L) \ 2: COL = (80 - L) \ 2
FOR I = 1 TO L
    LOCATE ROW + I, COL + I: PRINT MID$(A$, I, 1)
NEXT I
```

'1.4

' This program displays the numbers on the sides of a die.

,

```
INPUT "Enter number on top:"; T
INPUT "Enter number on front:"; F
INPUT "Enter number on right:"; R
PRINT "TOP="; T
PRINT "FRONT="; F
PRINT "RIGHT="; R
PRINT "BOTTOM="; 7 - T
PRINT "BACK="; 7 - F
PRINT "LEFT="; 7 - R
```

```
'1.5
' This program will fill the screen with random characters.
'
CLS
FOR I = 1 TO 24
  FOR J = 1 TO 80
    X = INT(RND(3) * 96) + 33
    PRINT CHR$(X);
  NEXT J
NEXT I
WHILE A$ = "": A$ = INKEY$: WEND
CLS
```

```
'1.6
' This program will display a rectangular array of periods.
'
INPUT "Enter coordinates:"; UR, UC, LR, LC
CLS
FOR I = UR TO LR
  FOR J = UC TO LC
    LOCATE I, J: PRINT ".";
  NEXT J
NEXT I
```

```
'1.7
' This program will generate 10 random numbers given a seed.
'
INPUT "Enter seed:"; SEED
FOR I = 1 TO 10
  RAND = (SEED * 421 + 1)
  RAND = RAND - INT(RAND / 100) * 100
  SEED = RAND
  PRINT RAND
NEXT I
```

```
'1.8
' This program will determine the mass of a fish tank.
'
INPUT "Enter K, L, W, H:"; K, L, W, H
MASS = L * 12 * 2.54 * W * 12 * 2.54 * H * 12 * 2.54
MASS = MASS / 1000 + K
PRINT USING "#####.## KILOGRAMS"; MASS;
```

```
'1.9
' This program will display 21 rows of letters.
,
CLS
FOR I = 1 TO 21
  IF I MOD 2 = 1 THEN
    PRINT STRING$(31, 64 + I)
  ELSE
    FOR J = 1 TO 10
      PRINT CHR$(64 + I); SPACE$(2);
    NEXT J
    PRINT CHR$(64 + I)
  END IF
NEXT I
```

```
'1.10
' This program will display the time needed to read a book.
,
DATA THE HISTORY OF THE COMPUTER,400
DATA THE RED DOG RUNS,200
DATA EATING APPLE PIE,150
DATA THE ART OF WINNING,250
INPUT "Enter book title:"; B$
INPUT "Enter rate (minutes/page):"; SP
I = 0
WHILE (I < 4) AND (A$ <> B$)
  READ A$, PA
  I = I + 1
WEND
M = PA * SP
H = INT(M / 60): M = M - H * 60
PRINT H; "HOURS "; M; "MINUTES"
```

'2.1

' This program will rotate a string N times to the left.

```
,
INPUT "Enter string: "; S$
INPUT "Enter N:"; N
L = LEN(S$)
N = N MOD L
PRINT RIGHT$(S$, L - N); LEFT$(S$, N)
```

'2.2

' This program will determine the number of diskettes bought.

```
,
FOR V = 1 TO 98
  FOR M = 1 TO 99 - V STEP 5
    W = 100 - V - M
    IF W >= 0 AND V * 225 + M * 297 + W * 120 = 23607 THEN
      PRINT V; "VERS "; M; "MAXS "; W; "WABS": END
    END IF
  NEXT M
NEXT V
```

'2.3

' This program will display a subset of random numbers.

```
,
DIM A(15)
RANDOMIZE TIMER
INPUT "Enter list item:"; ITEM
WHILE ITEM <> -1
  A(J) = ITEM
  INPUT "Enter list item:"; ITEM
  J = J + 1
WEND
WHILE A$ <> CHR$(27)
  FOR I = 0 TO 4
    SWAP A(I), A(INT(RND * (J - I) + I))
    PRINT A(I)
  NEXT I
  PRINT "PRESS ANY KEY": A$ = ""
  WHILE A$ = "": A$ = INKEY$: WEND
WEND
```

'2.4

' This program will display all partitioned sum of number.

,

```
INPUT "Enter a number less than 20: "; N
```

```
FOR I = N TO 1 STEP -1
```

```
  IF N MOD I = 0 THEN
```

```
    X = INT(N / I)
```

```
    PRINT SPACE$(N - X);
```

```
    I$ = MID$(STR$(I), 2)
```

```
    PRINT I$;
```

```
    IF I < N THEN
```

```
      FOR J = 1 TO X - 1
```

```
        PRINT "+"; I$;
```

```
        NEXT J
```

```
    END IF
```

```
    PRINT
```

```
  END IF
```

```
NEXT I
```

'2.5

' This program will calculate the fractional value.

,

```
INPUT "Enter word: "; A$
```

```
FOR I = 1 TO 3
```

```
  A(I) = ASC(MID$(A$, I, 1)) - 64
```

```
NEXT I
```

```
N = A(1) * A(2) + A(2) * A(3) + A(1) * A(3)
```

```
D = A(1) * A(2) * A(3)
```

```
FOR I = D TO 1 STEP -1
```

```
  IF N MOD I = 0 AND D MOD I = 0 THEN
```

```
    PRINT LTRIM$(STR$(N / I)); "/"; LTRIM$(STR$(D / I)): END
```

```
  END IF
```

```
NEXT I
```

```
'2.6
' This program will find a subset of integers.
'
I = 1
INPUT "Enter set item:"; ITEM(I)
WHILE ITEM(I) > 0
  I = I + 1
  INPUT "Enter set item:"; ITEM(I)
WEND
LASTI = I - 1
INPUT "Enter N:"; N
INPUT "Enter S:"; S
' Sort list
FOR I = 1 TO LASTI - 1
  FOR J = I + 1 TO LASTI
    IF ITEM(I) > ITEM(J) THEN SWAP ITEM(I), ITEM(J)
  NEXT J
NEXT I
SUM = 0
FOR I = 1 TO N: SUM = SUM + ITEM(I): NEXT I
IF SUM > S THEN PRINT "NO": END
PRINT " YES"
FOR I = 1 TO N: PRINT ITEM(I); : NEXT I
```

```
'2.7
' This program will determine if patterns are legal/illegal.
'
DATA 1,4,3,4,4,5
DATA 5,2,5,2,5,5
FOR I = 0 TO 5: READ A(I): NEXT I
FOR I = 0 TO 5: READ B(I): NEXT I
INPUT "Enter pattern:"; P$
STATE = 0
'
' Run the state machine
'
LP = LEN(P$)
FOR I = 1 TO LP      'Check whole string even if error found
  C$ = MID$(P$, I, 1)
  IF C$ <> "A" AND C$ <> "B" THEN STATE = 5 'illegal pattern
  IF C$ = "A" THEN STATE = A(STATE) ELSE STATE = B(STATE)
NEXT I
IF STATE = 4 THEN PRINT "LEGAL PATTERN": END
PRINT "ILLEGAL PATTERN"
```

'2.8

' This program will find integers having F factors.

```
'  
INPUT "Enter M, N, F:"; M, N, F  
FOR I = M TO N  
  S = 0: X = INT(SQR(I) + .000001)  
  FOR J = 1 TO X  
    IF I MOD J = 0 THEN S = S + 2  
  NEXT J  
  IF X * X = I THEN S = S - 1  
  IF S = F THEN PRINT I  
NEXT I
```

'2.9

' This program will alphabetize 5 words according to rules.

```
'  
DIM A$(12), B$(12), C$(12)  
FOR I = 1 TO 5  
  INPUT "Enter word: "; A$(I): L = LEN(A$(I))  
  FOR J = 1 TO L  
    B$(J) = MID$(A$(I), J, 1)  
  NEXT J  
  ' Alphabetize letters within word to make word2 (C$)  
  FOR J = 1 TO L - 1  
    FOR K = J + 1 TO L  
      IF B$(J) > B$(K) THEN SWAP B$(J), B$(K)  
    NEXT K  
    C$(I) = C$(I) + B$(J)  
  NEXT J  
  C$(I) = C$(I) + B$(L)  
NEXT I  
  ' Alphabetize words according to word2 (C$)  
FOR I = 1 TO 4  
  FOR J = I + 1 TO 5  
    IF C$(I) > C$(J) THEN SWAP C$(I), C$(J): SWAP A$(I), A$(J)  
  NEXT J  
NEXT I  
FOR I = 1 TO 5: PRINT A$(I): NEXT I
```

```

'2.10
' This program will produce a super-duper input routine
' with 4 types of input.
,
INPUT "Enter ROW, COL:"; ROW, COL
INPUT "Enter MAX:"; MAX
INPUT "Enter TYPE:"; TYP
CLS : CH$ = " ": INITCOL = COL
DO UNTIL CH$ = CHR$(13)
  LOCATE ROW, COL: CH$ = ""
  WHILE CH$ = "": CH$ = INKEY$: WEND
,
  IF CH$ = CHR$(8) THEN ' Backspace pressed
    IF LEN(ENTRY$) > 0 THEN
      ENTRY$ = LEFT$(ENTRY$, LEN(ENTRY$) - 1)
      COL = COL - 1: LOCATE ROW, COL: PRINT " ";
    END IF
  ELSE
    VALIDCH = LEN(ENTRY$) < MAX
    IF VALIDCH THEN
      SELECT CASE TYP
        CASE 1
          IF CH$ <> " " AND (CH$ < "A" OR CH$ > "Z") THEN VALIDCH = 0
        CASE 2
          IF CH$ <> "." AND (CH$ < "0" OR CH$ > "9") THEN VALIDCH = 0
        CASE 3
          IF COL - INITCOL = 2 OR COL - INITCOL = 5 THEN
            IF CH$ <> "-" THEN VALIDCH = 0
          ELSE
            IF CH$ < "0" OR CH$ > "9" THEN VALIDCH = 0
          END IF
        END SELECT
      END IF
    END IF
    IF VALIDCH THEN
      PRINT CH$;
      ENTRY$ = ENTRY$ + CH$
      COL = COL + 1
    END IF
  END IF
LOOP
LOCATE ROW + 2, INITCOL: PRINT ENTRY$

```

```
'3.1
' This program will determine if 2 words are closely spelled.
'
INPUT "Enter word 1: "; W1$
INPUT "Enter word 2: "; W2$
L1 = LEN(W1$): L2 = LEN(W2$)
IF ABS(L1 - L2) > 1 THEN PRINT "NOT CLOSE": END
' Find first position where words differ
IF L1 < L2 THEN MIN = L1 ELSE MIN = L2
J = 1
WHILE (J <= MIN) AND MID$(W1$, J, 1) = MID$(W2$, J, 1)
  J = J + 1
WEND
IF J > MIN THEN PRINT "CLOSE": END 'Equal or differ by ins/del
IF L1 = L2 THEN
' Check for transposition or one symbol change
  IF J <> L1 THEN
    IF MID$(W1$, J + 1, 1) = MID$(W2$, J, 1) THEN
      IF MID$(W2$, J + 1, 1) = MID$(W1$, J, 1) THEN
        J = J + 1 'Skip over possible transposition
      END IF
    END IF
  END IF
  IF MID$(W1$, J + 1) = MID$(W2$, J + 1) THEN PRINT "CLOSE": END
  PRINT "NOT CLOSE": END
ELSE
' Check for insertion or deletion
  IF L1 > L2 THEN
    IF MID$(W2$, J) = MID$(W1$, J + 1) THEN PRINT "CLOSE": END
    PRINT "NOT CLOSE"
  ELSE
    IF MID$(W1$, J) = MID$(W2$, J + 1) THEN PRINT "CLOSE": END
    PRINT "NOT CLOSE"
  END IF
END IF
```

```

'3.2
' This program will evaluate an NxN determinant for N=2,3,4.
,
INPUT "Enter dimension N:"; N
FOR I = 1 TO N
  FOR J = 1 TO N
    PRINT USING "Enter row #"; I;
    PRINT USING ", col #:"; J; : INPUT A(I, J)
  NEXT J
NEXT I
' -- 2x2
IF N = 2 THEN
  PRINT A(1, 1) * A(2, 2) - A(1, 2) * A(2, 1)
ELSE
' -- 3x3
  IF N = 3 THEN
    K = 4: GOSUB Det3x3
    PRINT T
  ELSE
' -- 4x4
    FOR K = 1 TO 4
      A = A(4, K) * (-1) ^ K
    ,
Det3x3:
    FOR I = 1 TO 3
      FOR J = 1 TO 4
        IF J <> K THEN
          S = S + 1: B(I, S) = A(I, J)
          B(I, S + 3) = A(I, J)
        END IF
      NEXT J: S = 0
    NEXT I
    FOR I = 1 TO 3
      T = T + B(1, I) * B(2, I + 1) * B(3, I + 2)
      T = T - B(1, I + 2) * B(2, I + 1) * B(3, I)
    NEXT I
    IF N = 3 THEN RETURN
  ,
    B = B + T * A: T = 0
  NEXT K: PRINT B
END IF
END IF

```

'3.3

' This program will display the number of word occurrences.
,

```
DIM WORD$(50), WORDTOT(50)
INPUT "Enter text: "; LINES$
START = 1: NUMOFWORDS = 0
WHILE START <= LEN(LINES$)
  ENDOFWORD = 0: NEXTWORD$ = ""
  WHILE (START <= LEN(LINES$)) AND (NOT ENDOFWORD)
    CH$ = MID$(LINES$, START, 1)
    IF (CH$ < "A" OR CH$ > "Z") AND (CH$ <> "'") THEN
      ENDOFWORD = -1
    ELSE
      NEXTWORD$ = NEXTWORD$ + CH$
    END IF
    START = START + 1
  WEND
  IF NEXTWORD$ > "" THEN NEWWORD = -1 ELSE NEWWORD = 0
  WORDIND = 0
  WHILE (WORDIND < NUMOFWORDS) AND NEWWORD
    WORDIND = WORDIND + 1
    IF NEXTWORD$ = WORD$(WORDIND) THEN NEWWORD = 0
  WEND
  IF NOT NEWWORD THEN
    WORDTOT(WORDIND) = WORDTOT(WORDIND) + 1
  ELSE
    ' Add new word to list
    NUMOFWORDS = NUMOFWORDS + 1
    WORD$(NUMOFWORDS) = NEXTWORD$
    WORDTOT(NUMOFWORDS) = 1
  END IF
WEND
FOR I = 1 TO NUMOFWORDS
  PRINT WORDTOT(I); WORD$(I)
NEXT I
```

```
'3.4
' This program will encrypt a string such that when this
' code is entered, the string will be reproduced.
'
DIM ASCI(30)
INPUT "Enter text: "; ST$
NUMOFCH = 0: I = 1
WHILE (I <= LEN(ST$))
  CH$ = MID$(ST$, I, 1): NUMOFCH = NUMOFCH + 1
  IF CH$ = "\" THEN
    I = I + 1: NEXTCH$ = MID$(ST$, I, 1)
    IF NEXTCH$ = "\" THEN
      ASCI(NUMOFCH) = ASC(NEXTCH$)
    ELSE
      ASCST$ = MID$(ST$, I, 3)
      ASCI(NUMOFCH) = VAL(ASCST$)
      I = I + 2
    END IF
  ELSE
    ASCI(NUMOFCH) = ASC(CH$) 'Regular character
  END IF
  I = I + 1
WEND
' Encrypt code
FOR I = 1 TO NUMOFCH
  CODENUM = 255 - ASCI(I)
  IF (CODENUM >= 32) AND (CODENUM <= 92) THEN
    PRINT CHR$(CODENUM);
    IF CODENUM = ASC("\" ) THEN PRINT "\";
  ELSE
    PRINT "\";
    PRINT MID$(STR$(1000 + CODENUM), 3, 3);
  END IF
NEXT I
```

```

'3.5
' This program will unscramble the numbers 5132, 4735, and
' 8014153 so that the first times the second equals the
' third with a missing digit.
'
DIM D(24), E(24)
DATA 5,1,3,2
DATA 4,7,3,5
DATA 8,0,1,4,1,5,3
FOR I = 1 TO 4: READ A(I): NEXT I
FOR I = 1 TO 4: READ B(I): NEXT I
FOR I = 1 TO 7: READ B$(I): NEXT I
FOR A = 1 TO 4
  FOR B = 1 TO 4
    FOR C = 1 TO 4: D = 10 - A - B - C
      D = 4 + 3 + 2 + 1 - A - B - C
      IF A <> B AND B <> C AND A <> C THEN
        S = S + 1
        D(S) = A(A) * 1000 + A(B) * 100 + A(C) * 10 + A(D)
        E(S) = B(A) * 1000 + B(B) * 100 + B(C) * 10 + B(D)
      END IF
    NEXT C
  NEXT B
NEXT A
FOR I = 1 TO 24
  FOR J = 1 TO 24
    X# = D(I) * E(J)
    A$ = LTRIM$(STR$(X#))
    IF LEN(A$) = 8 THEN
      FOR K = 1 TO 8
        A$(K) = MID$(A$, K, 1)
      NEXT K
      B = 1: MATCH = -1
      WHILE (B <= 7) AND MATCH
        MATCH = 0: A = 1
        WHILE (A <= 8) AND NOT MATCH
          IF B$(B) = A$(A) THEN A$(A) = " ": MATCH = -1
          A = A + 1
        WEND
      B = B + 1
    WEND
    IF MATCH THEN PRINT D(I); E(J); " "; A$
  END IF
NEXT J
NEXT I

```

```

'3.6
' This program will display the front colors on the Rubik's
' Pocket Cube after a move of T or F is performed.
'
DIM A$(24)
DATA W,Y,O,G,R,B
FOR I = 1 TO 6: READ A$
  FOR J = 1 TO 4
    S = S + 1: A$(S) = A$
  NEXT J
NEXT I
INPUT "Enter T, F, or Q: "; A$
DO UNTIL A$ = "Q"
  IF A$ = "T" THEN
    ' Rotate Top
    X$ = A$(1): A$(1) = A$(3): A$(3) = A$(4)
    A$(4) = A$(2): A$(2) = X$
    X$ = A$(5): A$(5) = A$(9): A$(9) = A$(13)
    A$(13) = A$(17): A$(17) = X$
    X$ = A$(6): A$(6) = A$(10): A$(10) = A$(14)
    A$(14) = A$(18): A$(18) = X$
  ELSE
    ' Rotate Front
    X$ = A$(5): A$(5) = A$(7): A$(7) = A$(8)
    A$(8) = A$(6): A$(6) = X$
    X$ = A$(3): A$(3) = A$(20): A$(20) = A$(22)
    A$(22) = A$(9): A$(9) = X$
    X$ = A$(4): A$(4) = A$(18): A$(18) = A$(21)
    A$(21) = A$(11): A$(11) = X$
  END IF
  ' Display front side
  PRINT A$(5); " "; A$(6)
  PRINT A$(7); " "; A$(8)
  INPUT "Enter T, F, or Q: "; A$
LOOP

```

```

'3.7
' This program will simulate a drill of adding Roman Numerals.
,
CLS
INPUT "Enter name: "; NME$
INPUT "Enter date: "; DAYTE$
,
DATA M,1000,D,500,C,100,L,50,X,10,V,5,I,1
FOR I = 1 TO 7: READ B$(I), B(I): NEXT I
,
DO UNTIL A$ = "3"
  CLS
  PRINT "1. INSTRUCTION PAGE"
  PRINT "2. PRACTICE 3 PROBLEMS"
  PRINT "3. QUIT"
  A$ = INPUT$(1)
  SELECT CASE A$
    CASE "1"
      CLS
      PRINT "YOU WILL BE GIVEN 3 PROBLEMS TO"
      PRINT "WORK. A PROBLEM WILL CONSIST OF"
      PRINT "ADDING TWO RANDOMLY GENERATED"
      PRINT "ROMAN NUMERALS LESS THAN 20."
      PRINT "YOU WILL TYPE YOUR ANSWER IN"
      PRINT "ROMAN NUMERALS AND PRESS 'RETURN.'"
      PRINT "(PRESS ANY KEY TO RETURN TO MENU.)"
      AN$ = INPUT$(1)
      ,
      ' Practice 3 problems
      ,
    CASE "2"
      RIGHT = 0: WRONG = 0
      FOR PROB = 1 TO 3
        CLS
        RANDOMIZE TIMER
        X(1) = INT(RND * 19) + 1: X(2) = INT(RND * 19) + 1
        X(3) = X(1) + X(2): HELP = X(3)
        FOR K = 1 TO 3: X$(K) = "": NEXT K
        FOR K = 1 TO 3
          FOR I = 1 TO 7
            X = X(K) / B(I)
            IF (ABS(X - 9 / 5) > .01) OR (I MOD 2 = 1) THEN
              X = INT(X)
              SELECT CASE X
                CASE 9
                  X$(K) = X$(K) + B$(I) + B$(I - 2)
                CASE 4
                  X$(K) = X$(K) + B$(I) + B$(I - 1)
                CASE IS > 0
                  FOR J = 1 TO X: X$(K) = X$(K) + B$(I): NEXT J
              END SELECT
              X(K) = X(K) - B(I) * X
            END IF
          NEXT I
        NEXT K
      ,
    CASE "3"
      CLS
      PRINT "THANK YOU FOR PLAYING"
      PRINT "PLEASE RESTART THE PROGRAM"
      PRINT "WHEN YOU WANT TO PLAY AGAIN"
      PRINT "PRESS ANY KEY TO RETURN TO MENU."
      AN$ = INPUT$(1)
      ,
      ' Practice 3 problems
      ,
  END SELECT
END DO

```

```

'
' Display problem
'
LOCATE 10, 15: PRINT X$(1): X = LEN(X$(1))
Y = LEN(X$(2)): COL = 15 + (X - Y) - 2
LOCATE 11, COL: PRINT "+ "; X$(2)
LOCATE 12, COL: PRINT STRING$(2 + Y, "-"): MISS = -1
WHILE MISS <> 0
  LOCATE 13, COL: INPUT N$
  '
  ' Evaluate
  '
  IF N$ = X$(3) THEN
    RIGHT = RIGHT + 1: MISS = 0
  ELSE
    IF MISS > 0 THEN
      MISS = 0: BEEP: WRONG = WRONG + 1: WR$(WRONG) = N$
      RI$(WRONG) = X$(3): RI(WRONG) = HELP
    ELSE
      MISS = 1: BEEP: LOCATE 16, COL: PRINT HELP
      LOCATE 13, COL: PRINT SPACE$(15)
    END IF
  END IF
WEND
NEXT PROB
'
' Progress Report
'
CLS : PRINT SPACE$(11); "PROGRESS REPORT"
PRINT "DATE: "; DAYTE$
PRINT "NAME: "; NME$
PRINT "NUMBER CORRECT: "; RIGHT
PRINT "NUMBER OF EXERCISES: 3"
PRINT "PERCENT CORRECT: "; INT(RIGHT / 3 * 100 + .5); "%"
PRINT
IF WRONG > 0 THEN
  LOCATE 15, 1: PRINT "WRONG ANSWER    CORRECT ANSWER    ARABIC"
  FOR I = 1 TO WRONG
    LOCATE 16 + I, 1: PRINT WR$(I)
    LOCATE 16 + I, 16: PRINT RI$(I)
    LOCATE 16 + I, 32: PRINT RI(I)
  NEXT I
END IF
LOCATE 23, 1: PRINT "PRESS ANY KEY TO RETURN TO MENU.";
AN$ = INPUT$(1)
END SELECT
LOOP

```

```
'3.8
' This program will determine the area shared w/2 rectangles.
'
DIM AB(20, 20), XY(20, 20)
FOR I = 1 TO 4
  INPUT "Enter X,Y: "; X(I), Y(I)
  X(I) = ABS(X(I)): Y(I) = ABS(Y(I))
NEXT I
FOR I = 1 TO 4
  INPUT "Enter A,B: "; A(I), B(I)
  A(I) = ABS(A(I)): B(I) = ABS(B(I))
NEXT I
'
' Store a 1 in each occupied square of AB
'
FOR I = A(1) TO A(2)
  FOR J = B(4) TO B(1)
    AB(I, J) = 1
  NEXT J
NEXT I
'
' Determine area in common (Heighth-1 x Width-1)
'
FOR I = X(1) TO X(2)
  FOR J = Y(4) TO Y(1)
    IF AB(I, J) = 1 THEN WDTN = WDTN + 1      'Both interior
  NEXT J
  IF WDTN > 0 THEN HEIGHT = HEIGHT + 1: WDTN2 = WDTN: WDTN = 0
NEXT I
PRINT (HEIGHT - 1) * (WDTN2 - 1)
```

```

'3.9
' This program will divide 2 big numbers with at most 30 digits.
'
DIM A(30), B(30), C(30), D(30)
INPUT "Enter first number: "; A$: LENA = LEN(A$)
INPUT "Enter second number: "; B$: LENB = LEN(B$)
L = LENB
'
' Store digits in arrays
'
FOR I = LENB TO 1 STEP -1
  B(LENB - I + 1) = VAL(MID$(B$, I, 1))
NEXT I
FOR I = LENB TO 1 STEP -1
  A(LENB - I + 1) = VAL(MID$(A$, I, 1))
NEXT I: K = LENB
'
' Shift digits of A until portion of A is greater than B
'
ShiftDigits:
  IF L <> LENB THEN
NextShift:
  K = K + 1: IF LENA < K THEN GOTO DisplayRemainder
  FOR I = L TO 1 STEP -1
    A(I + 1) = A(I)
  NEXT I
  A(1) = VAL(MID$(A$, K, 1))
  L = L + 1
  IF L < LENB THEN PRINT "0"; : GOTO NextShift
END IF
  IF L <= LENB THEN
  FOR I = LENB TO 1 STEP -1
    IF A(I) > B(I) THEN GOTO DivideAbyB
    IF A(I) <> B(I) THEN GOTO NextShift
  NEXT I
  ' All A(I) = B(I) at this point
END IF
'
' Divide A by B by subtracting B * J from A
'
DivideAbyB:
  SUBDONE = 0
  DO
    J = J + 1
    FOR I = 1 TO LENB
      C(I) = B(I) * J + C
      C = INT(C(I) / 10)
      C(I) = C(I) - C * 10
    NEXT I: C(I) = C: C = 0
    FOR I = 1 TO L
      D(I) = A(I) - C(I) - D
      D = -(D(I) < 0): IF D THEN D(I) = D(I) + 10
    NEXT I
    IF L - LENB = 0 OR D(L) = 0 THEN
      I = LENB + 1

```

```
    DO
      I = I - 1
      IF D(I) < B(I) THEN SUBDONE = -1
      LOOP UNTIL I = 1 OR D(I) > B(I) OR SUBDONE
    END IF
  LOOP UNTIL SUBDONE
  '
  ' Display J as # of subtractions done
  '
  PRINT USING "#"; J; : L = 0: J = 0
  FOR I = LENB TO 1 STEP -1
    IF D(I) > 0 OR T > 0 THEN
      T = 1: L = L + 1: A(I) = D(I)
    END IF
  NEXT I: T = 0: GOTO ShiftDigits
  '
  ' Display remainder
  '
DisplayRemainder:
  PRINT " Remainder ";
  FOR I = L TO 1 STEP -1
    PRINT USING "#"; A(I);
  NEXT I
  IF L = 0 THEN PRINT "0"
```

```

'3.10
' This program will generate random mazes with 3 x 5 paths.
'
CLS : RANDOMIZE TIMER: L = 8: W = 5
NUMOFLINES = (L - 1) * (W - 1) '# of lines to draw
LI = INT(32 / L)
WI = INT(15 / W)
LN = L: WN = W
DIM A(LN + 1, WN + 1) 'Points forbidden to start from
DIM PINT(33, 33) 'Existing points
'
' Draw perimeter
'
FOR I = 1 TO 33: PRINT "*"; : PINT(I - 1, 0) = 1: NEXT I
FOR I = 1 TO 14
  LOCATE I + 1, 1: PRINT "*": PINT(0, I) = 1
  LOCATE I + 1, 33: PRINT "*": PINT(L, I) = 1
NEXT I
FOR I = 1 TO 33: PRINT "*"; : PINT(I - 1, W) = 1: NEXT I
'
A(0, 0) = 1: A(LN, 0) = 1: A(LN, WN) = 1: A(0, WN) = 1
DO
  DO
    ' Get point that exists but is not forbidden
    X = INT(RND * 2 * LN) - INT(LN / 2)
    Y = INT(RND * 2 * WN) - INT(WN / 2)
    IF X < 0 THEN X = 0
    IF X > LN THEN X = LN
    IF Y < 0 THEN Y = 0
    IF Y > WN THEN Y = WN
  LOOP UNTIL (PINT(X, Y) = 1 AND A(X, Y) = 0)
  DO
    D = INT(RND * 4) 'Random direction
    SEGMENTDRAWN = 0: NUMOFTRIES = 0
    DO
      NUMOFTRIES = NUMOFTRIES + 1
      D = D + 1: IF D > 4 THEN D = D - 4
      SELECT CASE D
        '
        ' Up
        '
        CASE 1
          IF Y > 0 THEN
            IF PINT(X, Y - 1) = 0 THEN
              FOR J = 0 TO WI - 1
                LOCATE Y * WI - J, X * LI + 1: PRINT "*"
              NEXT J
              A = X: B = Y - 1: SEGMENTDRAWN = -1
            END IF
          END IF
        '
        ' Right
        '
        CASE 2
          IF X < LN THEN

```

```

        IF PINT(X + 1, Y) = 0 THEN
            FOR J = 0 TO LI - 1
                LOCATE Y * WI + 1, X * LI + 2 + J: PRINT "*"
            NEXT J
            A = X + 1: B = Y: SEGMENTDRAWN = -1
        END IF
    END IF
    '
    ' Down
    '
CASE 3
    IF Y < WN THEN
        IF PINT(X, Y + 1) = 0 THEN
            FOR J = 0 TO WI - 1
                LOCATE Y * WI + 2 + J, X * LI + 1: PRINT "*"
            NEXT J
            A = X: B = Y + 1: SEGMENTDRAWN = -1
        END IF
    END IF
    '
    ' Left
    '
CASE 4
    IF X > 0 THEN
        IF PINT(X - 1, Y) = 0 THEN
            FOR J = 0 TO LI - 1
                LOCATE Y * WI + 1, X * LI - J: PRINT "*"
            NEXT J
            A = X - 1: B = Y: SEGMENTDRAWN = -1
        END IF
    END IF
END SELECT
LOOP UNTIL SEGMENTDRAWN OR (NUMOFTRIES = 4)
'
IF SEGMENTDRAWN THEN
    PINT(A, B) = 1: LINESDRAWN = LINESDRAWN + 1
    X = A: Y = B
ELSE
    A(X, Y) = 1
END IF
LOOP UNTIL (LINESDRAWN = NUMOFLINES) OR NOT SEGMENTDRAWN
LOOP UNTIL (LINESDRAWN = NUMOFLINES)
'
' Open doors
'
X = INT(RND * WN) + 1: Y = INT(RND * WN) + 1
FOR J = 0 TO WI - 2
    LOCATE X * WI - J, 1: PRINT " "
    LOCATE Y * WI - J, 33: PRINT " "
NEXT J
LOCATE 23

```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '88
BASIC PROGRAM SOLUTIONS

```
'1.1
' This program clears the screen and prints a phrase
'
CLS
FOR I = 1 TO 10
  PRINT "THE BEST COMPUTER CONTEST!"
NEXT I

'1.2
' This program determines if a given input is integer or real.
'
INPUT "Enter #:"; NUM
IF NUM = INT(NUM) THEN PRINT "INTEGER" ELSE PRINT "REAL"

'1.3
' This program calculates the number of bytes on N diskettes.
'
INPUT "Enter N: "; N
PRINT N * 40 * 8 * 512

'1.4
' This program prints the computer component missing.
'
INPUT "Enter component:"; A$
INPUT "Enter component:"; B$
INPUT "Enter component:"; C$
INPUT "Enter component:"; D$
DATA CPU, PRIMARY, SECONDARY, INPUT, OUTPUT
FOR I = 1 TO 5
  READ E$
  IF NOT (A$ = E$ OR B$ = E$ OR C$ = E$ OR D$ = E$) THEN
    PRINT E$: END
  END IF
NEXT I
```

```
'1.5
' This program displays 4 rectangles of asterisks with #s.
,
CLS
FOR I = 1 TO 79: PRINT "*"; : NEXT I
FOR I = 2 TO 23
  LOCATE I, 1: PRINT "*"
  LOCATE I, 40: PRINT "*"
  LOCATE I, 79: PRINT "*"
NEXT I
LOCATE 12, 1: FOR I = 1 TO 79: PRINT "*"; : NEXT I
LOCATE 24, 1: FOR I = 1 TO 79: PRINT "*"; : NEXT I
' Place numbers in center of rectangles
LOCATE 6, 20: PRINT 1
LOCATE 6, 60: PRINT 2
LOCATE 18, 20: PRINT 3
LOCATE 18, 60: PRINT 4

'1.6
' This program displays the acronym for a given set of words.
,
INPUT "Enter words:"; A$
PRINT MID$(A$, 1, 1);
FOR I = 2 TO LEN(A$)
  MD$ = MID$(A$, I, 1)
  IF MD$ = " " THEN PRINT MID$(A$, I + 1, 1); : I = I + 1
NEXT I

'1.7
' This program will display 3 computer names in order of size.
,
INPUT "Enter name:"; N1$
INPUT "Enter type:"; T1$
INPUT "Enter name:"; N2$
INPUT "Enter type:"; T2$
INPUT "Enter name:"; N3$
INPUT "Enter type:"; T3$
IF T1$ = "MICRO" THEN PRINT N1$
IF T2$ = "MICRO" THEN PRINT N2$
IF T3$ = "MICRO" THEN PRINT N3$
IF T1$ = "MINI" THEN PRINT N1$
IF T2$ = "MINI" THEN PRINT N2$
IF T3$ = "MINI" THEN PRINT N3$
IF T1$ = "MAINFRAME" THEN PRINT N1$
IF T2$ = "MAINFRAME" THEN PRINT N2$
IF T3$ = "MAINFRAME" THEN PRINT N3$
```

```
'1.8
' This program will count the number of cans to be stacked.
'
INPUT "Enter N: "; N
FOR I = N TO 1 STEP -2
    SUM = SUM + I
NEXT I
PRINT SUM

'1.9
' This program simulates a queue w/options: ADD, TAKE, QUIT.
'
INPUT "Enter command:"; IN$
WHILE IN$ <> "QUIT"
    IF IN$ = "ADD" THEN
        MAX = MAX + 1: INPUT "Enter integer:"; A(MAX)
    ELSE
        IF IN$ = "TAKE" THEN
            MIN = MIN + 1: PRINT A(MIN)
        END IF
    END IF
    INPUT "Enter command:"; IN$
WEND

'1.10
' This program determines events of history between dates.
'
DATA 1642,"BLAISE PASCAL","ADDING MACHINE"
DATA 1801,"JOSEPH JACQUARD","PUNCHCARD AND WEAVING LOOM"
DATA 1830,"CHARLES BABBAGE","DESIGN OF ANALYTIC ENGINE"
DATA 1890,"HERMAN HOLLERITH","PUNCHCARD TABULATING MACHINE"
DATA 1944,"HOWARD AIKEN","MARK I"
DATA 1946,"ECKERT AND MAUCHLY","ENIAC"
DATA 1949,"VON NEUMAN","EDVAC"
'
INPUT "Enter years: "; Y1, Y2
FOR I = 1 TO 7
    READ DAT, NAM$, INV$
    IF Y1 <= DAT AND DAT <= Y2 THEN PRINT NAM$; " INVENTED "; INV$
NEXT I
```

```
'2.1
' This program displays a solid diamond of asterisks.
'
INPUT "Enter N: "; N
FOR I = 1 TO N STEP 2
  PRINT SPACE$( (N - I) / 2 );
  FOR J = 1 TO I: PRINT "*"; : NEXT J: PRINT
NEXT I
'
FOR I = N - 2 TO 1 STEP -2
  PRINT SPACE$( (N - I) / 2 );
  FOR J = 1 TO I: PRINT "*"; : NEXT J: PRINT
NEXT I

'2.2
' This program determines the efficiency order of 3 sorts.
'
INPUT "Enter N: "; N
B = N * (N - 1) / 2: B$ = "BUBBLE SORT"
S = N * (LOG(N) / LOG(2)) ^ 2: S$ = "SHELL SORT"
Q = N * (LOG(N) / LOG(2)): Q$ = "QUICK SORT"
'
IF B < S AND B < Q THEN
  PRINT B$
  IF S < Q THEN PRINT S$ ELSE PRINT Q$
  IF S < Q THEN PRINT Q$ ELSE PRINT S$
END
ELSE
  IF S < Q THEN
    PRINT S$
    IF B < Q THEN PRINT B$ ELSE PRINT Q$
    IF B < Q THEN PRINT Q$ ELSE PRINT B$
  END
  ELSE
    PRINT Q$
    IF B < S THEN PRINT B$ ELSE PRINT S$
    IF B < S THEN PRINT S$ ELSE PRINT B$
  END IF
END IF
```

'2.3

' This program determines the number of people in a group.

```

'
DEFINT A-Z
DIV(1) = 2: RE(1) = 1
DIV(2) = 3: RE(2) = 2
DIV(3) = 5: RE(3) = 1
DIV(4) = 7: RE(4) = 2
'
FOR NUM = 1 TO 200
  GOOD = -1
  FOR I = 1 TO 4
    IF NUM MOD DIV(I) <> RE(I) THEN GOOD = 0
  NEXT I
  IF GOOD THEN PRINT NUM: END
NEXT NUM

```

'2.4

' This program generates 5 random numbers between 0 and 9999.

```

'
INPUT "Enter seed:"; SEED
FOR I = 1 TO 5
  PROD# = SEED * SEED
  PROD$ = MID$(STR$(PROD#), 2)
  DIGITS = LEN(PROD$)
  IF DIGITS < 8 THEN
    ' **** Pad 0's to make 8 digit # ****
    FOR J = 1 TO 8 - DIGITS
      PROD$ = PROD$ + "0"
    NEXT J
  END IF
  SEED = VAL(MID$(PROD$, 3, 4))
  PRINT SEED
NEXT I

```

'2.5

' This program checks to see if data transmitted is Correct.

```

'
INPUT "Enter bits:"; BIT$
INPUT "Enter parity:"; PAR$
IF LEN(BIT$) <> 8 THEN PRINT "ERROR": END
FOR I = 1 TO 8
  MD$ = MID$(BIT$, I, 1)
  IF MD$ <> "0" AND MD$ <> "1" THEN PRINT "ERROR": END
  SUM = SUM + VAL(MD$)
NEXT I
' ERROR if even but odd parity; or if odd but even parity
IF SUM MOD 2 = 0 AND PAR$ <> "EVEN" THEN PRINT "ERROR": END
IF SUM MOD 2 = 1 AND PAR$ <> "ODD" THEN PRINT "ERROR": END
PRINT "CORRECT"

```

'2.6

' This program will calculate the area of a polygon.

```
'  
INPUT "Enter n: "; N  
FOR I = 1 TO N  
    INPUT "Enter vertex: "; X(I), Y(I)  
NEXT I  
'  
X(N + 1) = X(1): Y(N + 1) = Y(1)  
FOR I = 1 TO N  
    SUM = SUM + X(I) * Y(I + 1) - Y(I) * X(I + 1)  
NEXT I  
PRINT USING "AREA = ##.##"; ABS(SUM) / 2
```

'2.7

' This program displays the date before/after a given date.

```
'  
INPUT "Enter month, day, year: "; MONTH, DAY, YEAR  
DIM MO(12)  
FOR I = 1 TO 12: READ MO(I): NEXT I  
DATA 31,28,31,30,31,30,31,31,30,31,30,31  
'  
D1 = DAY - 1: D2 = DAY + 1: M1 = MONTH: M2 = MONTH  
Y1 = YEAR: Y2 = YEAR  
IF Y1 MOD 4 = 0 AND Y1 MOD 100 > 0 THEN LEAP = -1  
IF LEAP AND M1 = 3 AND D1 = 0 THEN LEAP1 = 1  
IF LEAP AND M2 = 2 AND D2 = 29 THEN LEAP2 = 1  
'  
IF D1 = 0 THEN  
    M1 = M1 - 1  
    IF M1 > 0 THEN D1 = MO(M1) + LEAP1  
    IF M1 = 0 THEN M1 = 12: D1 = MO(M1): Y1 = Y1 - 1  
ELSE  
    IF D2 > MO(M2) + LEAP2 THEN  
        M2 = M2 + 1: D2 = 1  
        IF M2 > 12 THEN M2 = 1: Y2 = Y2 + 1  
    END IF  
END IF  
'  
PRINT LTRIM$(STR$(M1));  
PRINT "-"; LTRIM$(STR$(D1)); "-"; LTRIM$(STR$(Y1))  
PRINT LTRIM$(STR$(M2));  
PRINT "-"; LTRIM$(STR$(D2)); "-"; LTRIM$(STR$(Y2))
```

```
'2.8
' This program displays a student's Cumulative G. P. Ave.
'
SEM = 1
WHILE SEM <= 8
  TOTAL = 0: HRSTOT = 0
  FOR I = 1 TO 4
    INPUT "Enter grade, credits:"; GR$, HRS
    POYNTS = 4 - (ASC(GR$) - 65)      'A=4 B=3 C=2 D=1 F=-1
    IF POYNTS = -1 THEN POYNTS = 0    'F=-1 becomes F=0
    TOTAL = TOTAL + POYNTS * HRS
    HRSTOT = HRSTOT + HRS
  NEXT I
'
GPA = TOTAL / HRSTOT
PRINT USING " GPA= #.###"; GPA
CUMTOTAL = CUMTOTAL + TOTAL: CUMHRS = CUMHRS + HRSTOT
CGPA = CUMTOTAL / CUMHRS
PRINT USING "CGPA= #.###"; CGPA
IF CGPA < 1 THEN DIS = -1
IF CGPA < 2 AND LASTCGPA < 2 AND SEM > 1 THEN DIS = -1
IF DIS THEN PRINT "STUDENT IS DISMISSED": END
LASTCGPA = CGPA
SEM = SEM + 1
WEND
```

```
'2.9
' This program displays 2 elements that form a battery.
'
DATA "LITHIUM " ,+3.05
DATA "SODIUM  " ,+2.71
DATA "ZINC    " ,+0.76
DATA "IRON   " ,+0.44
DATA "TIN    " ,+0.14
DATA "IODINE " , -0.54
DATA "SILVER " , -0.80
DATA "MERCURY" , -0.85
DATA "BROMINE" , -1.09
DATA "CHLORINE" , -1.36
FOR I = 1 TO 10: READ ELEM$(I), POT(I): NEXT I
'
INPUT "Enter Desired Voltage, Tolerance: "; VOLT, TOL
'
FOR I = 1 TO 10
  FOR J = 1 TO 10
    DIF = POT(I) - POT(J)
    IF DIF >= VOLT - TOL AND DIF <= VOLT + TOL THEN
      COUNT = COUNT + 1
      IF COUNT = 1 AND DISPLAY > 0 THEN
        PRINT "PRESS ANY KEY FOR MORE": A$ = INPUT$(1): PRINT
      END IF
      PRINT ELEM$(I); " "; ELEM$(J); " ";
      PRINT USING "#.##"; DIF
      DISPLAY = 1
    END IF
  IF COUNT = 8 THEN PRINT : COUNT = 0
  NEXT J
NEXT I
IF DISPLAY = 0 THEN PRINT "NO BATTERY CAN BE FORMED"
```

```

'2.10
' This program will keep score for a double dual race.
,
CLS : DIM IN$(21)
FOR I = 1 TO 21
  PRINT "Place "; I; ":"; : INPUT IN$(I)
  IF I > 1 THEN
    J = 1
    WHILE J <= TN AND INIT$(J) <> IN$(I): J = J + 1: WEND
  END IF
  IF (INIT$(J) <> IN$(I)) OR (I = 1) THEN
    TN = TN + 1: INIT$(TN) = IN$(I)
  END IF
NEXT I
' Assert TEAM$(1, 2, 3) = 3 unique team INITIALS
FOR I = 1 TO 2
  FOR J = I + 1 TO 3
    PL = 0: T1 = 0: T2 = 0: T1PL = 0: T2PL = 0
    FOR K = 1 TO 21
      IF IN$(K) = INIT$(I) THEN
        PL = PL + 1: T1 = T1 + PL: T1PL = T1PL + 1
        TEAM1(T1PL) = PL
      END IF
      IF IN$(K) = INIT$(J) THEN
        PL = PL + 1: T2 = T2 + PL: T2PL = T2PL + 1
        TEAM2(T2PL) = PL
      END IF
    NEXT K
    T1 = T1 - TEAM1(6) - TEAM1(7)
    T2 = T2 - TEAM2(6) - TEAM2(7)
    PRINT "TEAM "; INIT$(I); ":"; T1; " POINTS"
    PRINT "TEAM "; INIT$(J); ":"; T2; " POINTS"
    IF (T1 < T2) OR (T1 = T2 AND TEAM1(6) < TEAM2(6)) THEN
      PRINT "TEAM "; INIT$(I);
    ELSE
      PRINT "TEAM "; INIT$(J);
    END IF
    PRINT " WINS!": PRINT
  NEXT J
NEXT I

```

```
'3.1
' This program puts a set of real numbers in numerical order.
'
INPUT "Enter N: "; N
FOR I = 1 TO N
  INPUT "Enter #: "; A(I)
NEXT I
DATA 0,8,1,2,5,4,3,9,7,6
FOR I = 0 TO 9: READ PLACE: ORDER(PLACE) = I: NEXT I
'   ***  replace digits in duplicated number   ***
FOR I = 1 TO N
  NUM$ = STR$(A(I))
  FOR J = 1 TO LEN(NUM$)
    MD$ = MID$(NUM$, J, 1)
    NUM = VAL(MD$)
    IF NUM > 0 OR MD$ = "0" THEN
      NUM2 = ORDER(NUM)
      MID$(NUM$, J, 1) = MID$(STR$(NUM2), 2)
    END IF
  NEXT J
  B(I) = VAL(NUM$)
NEXT I
'   ***  sort according to numbers with replaced digits   ***
FOR I = 1 TO N - 1
  FOR J = I + 1 TO N
    IF B(I) > B(J) THEN SWAP B(I), B(J): SWAP A(I), A(J)
  NEXT J
NEXT I
FOR I = 1 TO N: PRINT LTRIM$(STR$(A(I))): NEXT I
```

```
'3.2
' This program displays total number of ways to make change.
'
DEFINT B-Z
INPUT "Enter AMOUNT: "; AMOUNT
MAXQ = INT(AMOUNT * 4)
MAXD = INT(AMOUNT * 10)
MAXN = INT(AMOUNT * 20)
FOR Q = 0 TO MAXQ
  FOR D = 0 TO MAXD - INT(2.5 * Q)
    FOR N = 0 TO MAXN - 5 * Q - 2 * D
      COUNT = COUNT + 1
    NEXT N
  NEXT D
NEXT Q
PRINT COUNT
```

'3.3

' This program determines if a point/box is inside a 2nd box.

```

INPUT "Enter point: "; PX, PY, PZ
INPUT "Enter cubel diagonal point1: "; C1X1, C1Y1, C1Z1
INPUT "Enter cubel diagonal point2: "; C1X2, C1Y2, C1Z2
INPUT "Enter cube2 diagonal point1: "; C2X1, C2Y1, C2Z1
INPUT "Enter cube2 diagonal point2: "; C2X2, C2Y2, C2Z2
A = C1X1: B = C1X2: GOSUB MinOfAandB: C1MINX = MIN
A = C1Y1: B = C1Y2: GOSUB MinOfAandB: C1MINY = MIN
A = C1Z1: B = C1Z2: GOSUB MinOfAandB: C1MINZ = MIN
A = C2X1: B = C2X2: GOSUB MinOfAandB: C2MINX = MIN
A = C2Y1: B = C2Y2: GOSUB MinOfAandB: C2MINY = MIN
A = C2Z1: B = C2Z2: GOSUB MinOfAandB: C2MINZ = MIN
A = C1X1: B = C1X2: GOSUB MaxOfAandB: C1MAXX = MAX
A = C1Y1: B = C1Y2: GOSUB MaxOfAandB: C1MAXY = MAX
A = C1Z1: B = C1Z2: GOSUB MaxOfAandB: C1MAXZ = MAX
A = C2X1: B = C2X2: GOSUB MaxOfAandB: C2MAXX = MAX
A = C2Y1: B = C2Y2: GOSUB MaxOfAandB: C2MAXY = MAX
A = C2Z1: B = C2Z2: GOSUB MaxOfAandB: C2MAXZ = MAX
'
PRINT "POINT ";
IF PX < C2MINX OR PY < C2MINY OR PZ < C2MINZ THEN
  PRINT "DOES NOT LIE";
ELSE
  IF PX > C2MAXX OR PY > C2MAXY OR PZ > C2MAXZ THEN
    PRINT "DOES NOT LIE";
  ELSE
    PRINT "LIES";
  END IF
END IF
PRINT " INSIDE 2ND CUBE"
'
PRINT "1ST CUBE ";
IF C1MINX < C2MINX OR C1MINY < C2MINY OR C1MINZ < C2MINZ THEN
  PRINT "DOES NOT LIE";
ELSE
  IF C1MAXX > C2MAXX OR C1MAXY > C2MAXY OR C1MAXZ > C2MAXZ THEN
    PRINT "DOES NOT LIE";
  ELSE
    PRINT "LIES";
  END IF
END IF
PRINT " INSIDE 2ND CUBE"
END
'*** SUBROUTINE to determine MIN of A and B
MinOfAandB:
  IF A <= B THEN MIN = A ELSE MIN = B
  RETURN
'*** SUBROUTINE to determine MAX of A and B
MaxOfAandB:
  IF A >= B THEN MAX = A ELSE MAX = B
  RETURN

```

```

'3.4
' This program produces an alphabetical list of permutations.
' **** Note: QBASIC has recursive capabilities, but this is
'         a way to do permutations without recursion.
' Also, this is an example of old BASIC (with line #s/branching).
'
10 DIM PERM$(720)
20 INPUT "Enter letters:"; A$: L = LEN(A$)
30 FOR I = 1 TO L: B$(I) = MID$(A$, I, 1): NEXT I: I = L
40 ON I GOTO 20, 90, 80, 70, 60, 50
50 FOR N6 = 1 TO 6: H = 5: GOSUB 340
60   FOR N5 = 1 TO 5: H = 4: GOSUB 340
70     FOR N4 = 1 TO 4: H = 3: GOSUB 340
80       FOR N3 = 1 TO 3: H = 2: GOSUB 340
90         FOR N2 = 1 TO 2
100          SWAP B$(I), B$(I - 1): TOTAL = TOTAL + 1
110          FOR J = 1 TO L
120            PERM$(TOTAL) = PERM$(TOTAL) + B$(J)
130          NEXT J
140        NEXT N2: IF I = 2 THEN 190
150      NEXT N3: IF I = 3 THEN 190
160    NEXT N4: IF I = 4 THEN 190
170  NEXT N5: IF I = 5 THEN 190
180 NEXT N6
190 '***  INSERTION SORT  ***
200 FOR I = 2 TO TOTAL
210   IND = I
220   WHILE PERM$(IND) < PERM$(IND - 1) AND IND > 1
230     SWAP PERM$(IND), PERM$(IND - 1): IND = IND - 1
240   WEND
250 NEXT I
260 '
270 FOR I = 1 TO TOTAL
280   IF PERM$(I) = PERM$(I - 1) THEN 300
290   PRINT PERM$(I): TOTAL2 = TOTAL2 + 1
300 NEXT I
310 PRINT "TOTAL="; TOTAL2
320 END
330 ' *****  SUBROUTINE  *****
340 Z$ = B$(I - H)
350 FOR J = I - H TO I - 1
360   B$(J) = B$(J + 1)
370 NEXT J
380 B$(I) = Z$
390 RETURN

```

'With QBASIC,
'<==This can be written
'using IF/END IF
'instead of branching.

```
'3.5
' This program will control the movements of a snake.
,
CLS : DIM A(25, 81)
V = 12: H = 8: LOCATE V, H
FOR I = 8 TO 32
  PRINT "*"; : A(V, I) = 1
  A$ = A$ + "12": B$ = B$ + RIGHT$(STR$(I), 2)
NEXT I
WHILE D$ = "": D$ = INKEY$: WEND: C$ = D$
DO UNTIL C$ = CHR$(27)
  FOR I = 1 TO 100
    D$ = INKEY$: IF D$ <> "" THEN C$ = D$
  NEXT I
  IF C$ = "I" THEN V = V - 1
  IF C$ = "M" THEN V = V + 1
  IF C$ = "J" THEN H = H - 1
  IF C$ = "K" THEN H = H + 1
  IF A(V, H) OR V = 0 OR V = 25 OR H = 0 OR H = 81 THEN END
  A(V, H) = 1: LOCATE V, H: PRINT "*"
  X = VAL(RIGHT$(A$, 2)): Y = VAL(RIGHT$(B$, 2))
  LOCATE X, Y: PRINT " "
  A(X, Y) = 0
  A$ = LEFT$(A$, 24 * 2): B$ = LEFT$(B$, 24 * 2)
  A$ = RIGHT$(STR$(V), 2) + A$
  B$ = RIGHT$(STR$(H), 2) + B$
LOOP
```

```

'3.6
' This program will solve two linear equations.
'
INPUT "Enter equation 1: "; E1$
INPUT "Enter equation 2: "; E2$
'
' Determine coefficients A1,B1,C1 and A2,B2,C2
'
EQ$ = E1$: ST = 1: GOSUB ParseEq: : A1 = VAAL
EQ$ = E1$: GOSUB ParseEq: : B1 = VAAL
EQ$ = E1$: GOSUB ParseEq: : C1 = VAAL
EQ$ = E2$: ST = 1: GOSUB ParseEq: : A2 = VAAL
EQ$ = E2$: GOSUB ParseEq: : B2 = VAAL
EQ$ = E2$: GOSUB ParseEq: : C2 = VAAL
'
' Compute solution if it exists
'
DEN = A1 * B2 - A2 * B1
NUMX = C1 * B2 - C2 * B1
NUMY = A1 * C2 - A2 * C1
IF DEN = 0 THEN PRINT "NO UNIQUE SOLUTION EXISTS.": END
PRINT "XSOLUTION= ";
IF NUMX / DEN < 0 THEN
  PRINT USING "##.#"; NUMX / DEN;
ELSE
  PRINT USING "#.#"; NUMX / DEN;
END IF
PRINT "  YSOLUTION= ";
IF NUMY / DEN < 0 THEN
  PRINT USING "##.#"; NUMY / DEN
ELSE
  PRINT USING "#.#"; NUMY / DEN
END IF
END
'
' Find Starting position ST of value
'
ParseEq:
  SYGN = 1      'Default to 1 (positive for unsigned #s)
  MD$ = "="
  WHILE MD$ = "="
    MD$ = MID$(EQ$, ST, 1)
    IF MD$ = "X" THEN VAAL = 1: ST = ST + 1: RETURN
    IF MD$ = "-" THEN ST = ST + 1
  WEND
  IF MD$ = "+" THEN ST = ST + 1
  IF MD$ = "-" THEN SYGN = -1: ST = ST + 1
'
' Find ending position EN of value
'
EN = ST: VAAL = 0: MD$ = MID$(EQ$, EN, 1): L = LEN(EQ$)
WHILE EN <= L AND (MD$ <> "X" AND MD$ <> "Y" AND MD$ <> "=")
  MD$ = MID$(EQ$, EN, 1)
  EN = EN + 1
WEND

```

```

EN = EN - 1
IF MD$ = "X" OR MD$ = "Y" OR MD$ = "=" THEN EN = EN - 1
IF MD$ = "=" THEN SYGN = -SYGN      'Bring C to other side
IF ST > EN THEN      'No Value
    VAAL = SYGN: ST = ST + 1
ELSE                  'Determine Value
    MD$ = MID$(EQ$, ST, EN - ST + 1)
    VAAL = SYGN * VAL(MD$): ST = EN + 2
END IF
RETURN

```

'3.7

' This program displays all semi-perfect #s between 2 and 35.

```

'
DEFINT A-Z
DIM A(20), B(20)
PRINT "SEMI #   EXAMPLE(S) "
FOR NUM = 2 TO 34: MAX = 0
    FOR DIV = 1 TO NUM / 2
        IF NUM MOD DIV = 0 THEN MAX = MAX + 1: B(MAX) = DIV
    NEXT DIV
    FOR B = 2 TO MAX
        L = MAX: GOSUB Combo
    NEXT B
NEXT NUM: END
'
' Produce combinations
'
Combo:
FOR I = 1 TO B: A(I) = B - I + 1: NEXT I
A(1) = A(1) - 1: N = 1
'
WHILE N <= B
    A(N) = A(N) + 1
    FOR I = N - 1 TO 1 STEP -1: A(I) = A(I + 1) + 1: NEXT I
    IF A(N) <= L - N + 1 THEN
        SUM = 0: FOR I = 1 TO B: SUM = SUM + B(A(I)): NEXT I
        IF SUM = NUM THEN
            PRINT USING "##"; NUM; : PRINT SPACE$(5); B(A(B));
            FOR I = B - 1 TO 1 STEP -1
                PRINT "+"; B(A(I));
            NEXT I: PRINT
        END IF
    END IF
    N = 0
END IF
N = N + 1
WEND
RETURN

```

```

'3.8
' This program will keep score for a bowler.
'
DIM A(10, 3): CLS
INPUT "Enter frames:"; F$: F$ = F$ + " "
FOR I = 1 TO 10
  COMMAPOS = INSTR(F$, " ")
  A$(I) = MID$(F$, 1, COMMAPOS - 1)
  F$ = MID$(F$, COMMAPOS + 1, LEN(F$) - COMMAPOS)
NEXT I
PRINT
PRINT "-1- -2- -3- -4- -5- -6- -7- -8- -9- -10-"
PRINT "----!----!----!----!----!----!----!----!----!"
FOR I = 1 TO 10
  PRINT SPACE$(3 - LEN(A$(I))); A$(I); "!";
NEXT I
PRINT
'
' Assign values to A Frames according to X, /, or pins
'
FOR FR = 1 TO 10
  L = LEN(A$(FR))
  FOR J = 1 TO L
    MD$ = MID$(A$(FR), J, 1)
    IF MD$ = "X" THEN
      A(FR, J) = 10: LOOK(FR) = 2
    ELSE
      IF MD$ = "/" THEN
        A(FR, J) = 10 - A(FR, J - 1): LOOK(FR) = 1
      ELSE
        A(FR, J) = VAL(MD$)
      END IF
    END IF
  NEXT J
NEXT FR
'
' Determine FFrame values with LOOK ahead
'
FOR FR = 1 TO 10
  SUM(FR) = SUM(FR - 1) + A(FR, 1) + A(FR, 2)
  IF LOOK(FR) > 0 THEN
    IF LOOK(FR) <= 1 THEN
      ' *** A spare / needs 1 more value added ***
      IF FR = 10 THEN
        SUM(FR) = SUM(FR) + A(FR, 3)
      ELSE
        SUM(FR) = SUM(FR) + A(FR + 1, 1)
      END IF
    ELSE
      ' *** A strike X needs 2 more values added ***
      IF FR = 10 THEN
        SUM(FR) = SUM(FR) + A(FR, 3)
      ELSE
        SUM(FR) = SUM(FR) + A(FR + 1, 1) + A(FR + 1, 2)
        IF FR <> 9 THEN

```

```

        IF A(FR + 1, 1) = 10 THEN
            SUM(FR) = SUM(FR) + A(FR + 2, 1)
        END IF
    END IF
END IF
END IF
END IF
END IF
'    *** Print FFrame's value ***
SUM$ = MID$(STR$(SUM(FR)), 2)
PRINT SUM$; SPACE$(3 - LEN(SUM$)); "!";
NEXT FR
PRINT : PRINT STRING$(40, "-")

```

'3.9

```

' This program will convert a real from one base to another.
'
INPUT "Enter M, N, #: "; M, N, NUM$
PRINT LEFT$(NUM$, 2);
NUM$ = MID$(NUM$, 3):
MDIGITS = LEN(NUM$) 'Digits on right of period(.)
'
NDIGITS = 1
WHILE (1 / N) ^ NDIGITS > (1 / M) ^ MDIGITS AND NDIGITS < 7
    NDIGITS = NDIGITS + 1
WEND
'
' SUM= Base 10 # of NUM$
'
FOR I = 1 TO MDIGITS
    MD$ = MID$(NUM$, I, 1)
    MD = ASC(MD$) - 48: IF MD > 9 THEN MD = MD - 7
    SUM = SUM + MD / (M ^ I)
NEXT I
'
' Convert base 10 decimal to Base N fraction
'
FOR I = 1 TO NDIGITS + 1
    SUM = SUM * N: NUM(I) = INT(SUM): SUM = SUM - NUM(I)
NEXT I
'
' Print fraction with last digit rounded according to NDIGIT+1
'
FOR I = 1 TO NDIGITS - 1
    PRINT CHR$(48 + NUM(I) - (NUM(I) > 9) * 7);
NEXT I
IF NUM(NDIGITS + 1) >= N / 2 THEN NUM(NDIGITS) = NUM(NDIGITS) + 1
PRINT CHR$(48 + NUM(NDIGITS) - (NUM(NDIGITS) > 9) * 7);

```

```

'3.10
' This program computes the composition of P(Q) and Q(P).
'
INPUT "Enter to the ORDER of p(x): "; PORDER
FOR I = PORDER TO 0 STEP -1
  PRINT "Enter coefficient for x**"; I; ": "; : INPUT PCO(I)
NEXT I: PRINT
INPUT "Enter to the ORDER of q(x): "; QORDER
FOR I = QORDER TO 0 STEP -1
  PRINT "Enter coefficient for x**"; I; ": "; : INPUT QCO(I)
NEXT I
PRINT "P(Q(X))= "; : GOSUB CompPofQ
PRINT
' ***** Swap P and Q to perform Q(P(X)) *****
SWAP PORDER, QORDER
IF PORDER > QORDER THEN MAX = PORDER ELSE MAX = QORDER
FOR I = 0 TO MAX: SWAP PCO(I), QCO(I): NEXT I
PRINT "Q(P(X))= "; : GOSUB CompPofQ
END
'
' ***** Compute composition P of Q *****
'
CompPofQ:
  COMPORDER = PORDER * QORDER
  FOR I = 1 TO COMPORDER: POFQ(I) = 0: NEXT I
  FOR I = 0 TO PORDER
    IF PCO(I) <> 0 THEN
      IF I = 0 THEN
        POFQ(0) = PCO(0)
      ELSE
        FOR J = 0 TO QORDER: PROD(J) = QCO(J): NEXT J
        PRODORDER = QORDER
        IF I <> 1 THEN
          FOR IN = 1 TO I - 1
            FOR J = 0 TO PRODORDER: PROD2(J) = 0: NEXT J
            FOR J = 0 TO PRODORDER
              FOR K = 0 TO QORDER
                PROD2(J + K) = PROD2(J + K) + PROD(J) * QCO(K)
              NEXT K
            NEXT J
            PRODORDER = J + K
            FOR L = 0 TO PRODORDER
              PROD(L) = PROD2(L): PROD2(L) = 0
            NEXT L
          NEXT IN
        END IF
        FOR J = 0 TO PRODORDER
          PROD(J) = PROD(J) * PCO(I)
        NEXT J
        FOR J = PRODORDER TO 0 STEP -1
          POFQ(J) = POFQ(J) + PROD(J)
        NEXT J
      END IF
    END IF
  NEXT I

```

```
' ***** Print composition *****  
FOR I = COMPORDER TO 0 STEP -1  
  IF I < COMPORDER THEN PRINT " + ";  
  PRINT LTRIM$(STR$(POFQ(I))); "X**"; LTRIM$(STR$(I));  
NEXT I  
RETURN
```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '89
BASIC PROGRAM SOLUTIONS

'1.1

' This program will print an indented phrase on each line.

'

```
CLS : P$ = "1989 COMPUTER CONTEST"  
FOR I = 1 TO 22: PRINT SPACE$(I); P$: NEXT I
```

'1.2

' This program will translate gigabytes to megabytes.

'

```
INPUT "Enter number of gigabytes:"; G  
PRINT G * 1024; "MEGABYTES"
```

'1.3

' This program displays a word in a backward-L format.

'

```
INPUT "Enter word:"; A$  
L = LEN(A$)  
FOR I = 1 TO L - 1  
    PRINT SPACE$(L - I); MID$(A$, I, 1)  
NEXT I  
PRINT A$
```

'1.4

' This program prints a pattern of numbers in pyramid form.

'

```
INPUT "Enter N:"; N  
FOR I = 1 TO N  
    PRINT SPACE$(10 - I); : PRINT USING "#"; I;  
    IF I > 1 THEN PRINT SPACE$(I * 2 - 3); : PRINT USING "#"; I;  
    PRINT  
NEXT I
```

'1.5

' This program corrects dates with A.D. or B.C.

'

```
INPUT "Enter date: "; D  
INPUT "Enter A.D. or B.C.: "; A$  
IF A$ = "B.C." AND D > 4 THEN PRINT D - 4; "B.C.": END  
IF A$ = "B.C." THEN PRINT 5 - D; "A.D.": END  
PRINT D + 4; "A.D"
```

```
'1.6
' This program will allow a user access with a password.
,
INPUT "ENTER PASSWORD:"; PSW$
I = 0
WHILE PSW$ <> "ITSME" AND I < 2
  PRINT "INVALID PASSWORD"
  INPUT "ENTER PASSWORD:"; PSW$
  I = I + 1
WEND
IF PSW$ = "ITSME" THEN
  PRINT "YOU HAVE ACCESS"
ELSE
  PRINT "YOU ARE TRESPASSING"
END IF

'1.7
' This program will display the best DBMS.
,
INPUT "Enter N: "; N: MAX = 0
FOR I = 1 TO N
  INPUT "Enter DBMS name: "; D$
  INPUT "Enter convenience, efficiency:"; C, E
  IF C + E > MAX THEN MAX = C + E: NM$ = D$
NEXT I
PRINT NM$; " IS BEST"

'1.8
' This program displays the unique elements of a list.
,
INPUT "Enter #:"; N: NUM = 0
WHILE N <> -999
  I = 1
  WHILE I <= NUM AND N <> A(I)
    I = I + 1
  WEND
  IF I > NUM THEN NUM = I: A(I) = N
  INPUT "Enter #:"; N
WEND
FOR I = 1 TO NUM: PRINT LTRIM$(STR$(A(I))); " "; : NEXT I
PRINT

'1.9
' This program determines how many feet deep of dollar coins
' over Texas is equivalent to a given probability.
,
INPUT "Enter probability:"; PROB
DOLVOL = 1.5 * 1.5 * 3 / 32: TEXASAREA = 262134
TEXASVOL = TEXASAREA * 5280 * 12 * 5280 * 12
INCHDEEP = (PROB / (TEXASVOL / DOLVOL))
PRINT INT(INCHDEEP / 12 + .5); "FEET DEEP"
```

```
'1.10
' This program will map a logical address to the physical.
,
B(0) = 219:  L(0) = 600
B(1) = 2300: L(1) = 14
B(2) = 90:   L(2) = 100
B(3) = 1327: L(3) = 580
B(4) = 1952: L(4) = 96
INPUT "Enter Seg#, Address: "; S, A
WHILE S <= 4
  IF A > L(S) THEN
    PRINT "ADDRESSING ERROR"
  ELSE
    PRINT B(S) + A
  END IF
  INPUT "Enter Seg#, Address: "; S, A
WEND
```

```
'2.1
' This program prints F(x) for a recursive function given x.
,
INPUT "Enter x:"; X
F(1) = 1: F(2) = 1: F(3) = 1
I = 3
WHILE I < X
  F(I + 1) = (F(I) * F(I - 1) + 2) / F(I - 2)
  I = I + 1
WEND
PRINT "F("; : PRINT USING "#"; X; : PRINT ")="; F(X)
```

```
'2.2
' This program will print the prime factors of a number.
,
INPUT "Enter #:"; NUM
WHILE NUM > 1
  I = 2
  WHILE (NUM MOD I) > 0
    I = I + 1
  WEND
  PRINT I;
  NUM = INT(NUM / I)
  IF NUM > 1 THEN PRINT "X";
WEND
```

```
'2.3
' This program will display a word without its vowels.
,
INPUT "Enter word:"; WORD$
VOW$ = "AEIOU"
FOR I = 1 TO LEN(WORD$)
  CH$ = MID$(WORD$, I, 1)
  IF INSTR(VOW$, CH$) = 0 THEN PRINT CH$;
NEXT I
```

```
'2.4
' This program produces the shortest possible identifiers.
'
FOR I = 1 TO 6
  INPUT "Enter name: "; A$(I)
NEXT I
FOR I = 1 TO 6
  K = 1: S$ = LEFT$(A$(I), 1)
  FOR J = 1 TO 6
    WHILE (I <> J) AND S$ = MID$(A$(J), 1, K) AND (K < LEN(A$(I)))
      K = K + 1
      S$ = S$ + MID$(A$(I), K, 1)
    WEND
  NEXT J
  PRINT S$
NEXT I
```

```
'2.5
' This program prints the # of distinguishable permutations.
'
DIM LETTER(26)
INPUT "Enter word:"; WORD$: L = LEN(WORD$)
' Calculate L factorial (assuming all different letters)
NUM = 1
FOR I = 1 TO L: NUM = NUM * I: NEXT I
' Divide out of Num the factorials of the same letters
FOR I = 1 TO L
  LETPOS = ASC(MID$(WORD$, I, 1)) - 64
  LETTER(LETPOS) = LETTER(LETPOS) + 1
  IF LETTER(LETPOS) > 1 THEN NUM = NUM / LETTER(LETPOS)
NEXT I
PRINT NUM
```

```
'2.6
' This program underlines parts of a sentence between 2 *'s.
'
INPUT "Enter sentence:"; SENT$
CLS : PRINT SENT$
UNDER = 0: COL = 0
FOR I = 1 TO LEN(SENT$)
  CH$ = MID$(SENT$, I, 1)
  IF CH$ = "*" THEN
    UNDER = NOT UNDER
  ELSE
    COL = COL + 1
    LOCATE 3, COL: PRINT CH$
    IF UNDER THEN LOCATE 4, COL: PRINT "-"
  END IF
NEXT I
PRINT
```

```

'2.7
' This program will compute an expression containing + - * /.
,
INPUT "Enter expression:"; ST$: NUMST$ = ""
' Parse first number in Num1 and second number in Num2
FOR I = 1 TO LEN(ST$)
  CH$ = MID$(ST$, I, 1)
  IF INSTR("+-*/", CH$) > 0 THEN
    SYMBOL$ = CH$: NUM1 = VAL(NUMST$): NUMST$ = ""
  ELSE
    NUMST$ = NUMST$ + CH$
  END IF
NEXT I
NUM2 = VAL(NUMST$)
IF SYMBOL$ = "+" THEN PRINT NUM1 + NUM2
IF SYMBOL$ = "-" THEN PRINT NUM1 - NUM2
IF SYMBOL$ = "*" THEN PRINT NUM1 * NUM2
IF SYMBOL$ = "/" THEN PRINT NUM1 / NUM2

```

```

'2.8
' This program will display the saddle point of a matrix.
,
DIM MAT(5, 5)
INPUT "Enter # Rows, # Cols:"; ROWS, COLS
FOR I = 1 TO ROWS
  FOR J = 1 TO COLS
    PRINT USING "Enter Row#"; I;
    PRINT USING " Col#"; J;
    INPUT MAT(I, J)
  NEXT J
NEXT I
' Find value smallest in row, largest in column
FOR I = 1 TO ROWS
  FOR J = 1 TO COLS
    SMALL = -1
    FOR K = 1 TO COLS
      IF (K <> J) AND (MAT(I, J) >= MAT(I, K)) THEN SMALL = 0
    NEXT K
    IF SMALL THEN
      LARGE = -1
      FOR K = 1 TO ROWS
        IF (K <> I) AND (MAT(I, J) <= MAT(K, J)) THEN LARGE = 0
      NEXT K
      IF LARGE THEN
        PRINT "SADDLE POINT ="; MAT(I, J); "AT ROW"; I;
        PRINT "COL"; J
      END IF
    END IF
  NEXT J
NEXT I

```

```
'2.9
' This program will sort a set of dates in increasing order.
,
DIM MO$(12)
DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY,AUGUST
DATA SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER
FOR I = 1 TO 12: READ MO$(I): NEXT I
INPUT "Enter # of dates:"; N
FOR I = 1 TO N
  INPUT "Enter month:"; M$(I)
  INPUT "Enter day: "; D(I)
  INPUT "Enter year: "; Y(I)
  PRINT
' Combine year, month, day (in that order) for sorting
  J = 1
  WHILE (J < 13) AND (M$(I) <> MO$(J)): J = J + 1: WEND
  SORT(I) = ((Y(I) * 100) + J) * 100 + D(I)
  INDEX(I) = I
NEXT I
' Sort dates according to values in Sort() and swap index()
FOR I = 1 TO N - 1
  FOR J = I + 1 TO N
    IF SORT(INDEX(I)) > SORT(INDEX(J)) THEN
      SWAP INDEX(I), INDEX(J)
    END IF
  NEXT J
NEXT I
FOR I = 1 TO N
  PRINT M$(INDEX(I)); D(INDEX(I)); Y(INDEX(I))
NEXT I
```

```

'2.10
' This program displays class grades and the averages.
,
DIM QIZ(5, 4)
DATA "D. WOOLY", "M. SMITH", "C. BROWN", "R. GREEN", "T. STONE"
FOR I = 1 TO 5: READ NAM$(I): NEXT I
DATA 100,92,90,90, 55,75,70,65, 94,70,62,70
DATA 90,74,80,85, 85,98,100,70
FOR I = 1 TO 5
  FOR J = 1 TO 4
    READ QIZ(I, J)
  NEXT J
NEXT I
FOR SCR = 1 TO 2
  CLS
  IF SCR = 2 THEN
    PRINT "          MS. HEINDEL'S MUSIC CLASS"
    PRINT "                FINAL GRADES"
    PRINT "                SPRING 1989"
    PRINT
  END IF
  PRINT "  NAME          Q1          Q2          Q3          Q4";
  IF SCR = 2 THEN PRINT "          AVERAGE" ELSE PRINT
  PRINT
,
  FOR I = 1 TO 5
    PRINT NAM$(I); : SUM = 0
    FOR J = 1 TO 4
      PRINT SPACE$(4); : PRINT USING "###"; QIZ(I, J);
      SUM = SUM + QIZ(I, J)
    NEXT J
    IF SCR = 2 THEN PRINT USING "    ###.##"; SUM / 4 ELSE PRINT
  NEXT I
  PRINT
  IF SCR = 1 THEN
    PRINT "Enter 5 grades for quiz 4:";
    INPUT QIZ(1, 4), QIZ(2, 4), QIZ(3, 4), QIZ(4, 4), QIZ(5, 4)
  END IF
NEXT SCR
' Display Column averages and class average
PRINT "AVERAGE:"; : TOTAL = 0
FOR I = 1 TO 4
  SUM = 0
  FOR J = 1 TO 5: SUM = SUM + QIZ(J, I): NEXT J
  PRINT USING " ###.##"; SUM / 5;
  TOTAL = TOTAL + SUM
NEXT I
PRINT : PRINT
PRINT USING "CLASS AVERAGE:###.##"; TOTAL / 20

```

```

'3.1
' This program will determine if a word is correctly spelled.
'
INPUT "Enter word:"; ST$
L = LEN(ST$): CORRECT = -1
'-- Check for E before suffixes ING, IBLE, ABLE
IF L >= 4 THEN
  PART$ = MID$(ST$, L - 2, 3)
  IF PART$ = "ING" AND MID$(ST$, L - 3, 1) = "E" THEN CORRECT = 0
END IF
IF L >= 5 THEN
  PART$ = MID$(ST$, L - 3, 4)
  IF PART$ = "IBLE" AND MID$(ST$, L - 4, 1) = "E" THEN CORRECT = 0
  IF PART$ = "ABLE" AND MID$(ST$, L - 4, 1) = "E" THEN CORRECT = 0
END IF
'-- Check if IE after C.
PART$ = ST$: I = INSTR(PART$, "IE")
WHILE (I > 0) AND CORRECT
  I = I - 1
  IF I >= 1 THEN IF MID$(PART$, I, 1) = "C" THEN CORRECT = 0
  PART$ = MID$(PART$, I + 3, LEN(PART$) - (I + 2))
  I = INSTR(PART$, "IE")
WEND
'-- Check if EI not after C.
PART$ = ST$: I = INSTR(PART$, "EI")
WHILE (I > 0) AND CORRECT
  CORRECT = 0
  IF I >= 2 THEN IF MID$(PART$, I - 1, 1) = "C" THEN CORRECT = -1
  PART$ = MID$(PART$, I + 3, LEN(PART$) - (I + 2))
  I = INSTR(PART$, "EI")
WEND
'-- Check for 3 consecutive same letters
I = 1
WHILE (I <= L - 2) AND CORRECT
  IF MID$(ST$, I, 1) = MID$(ST$, I + 1, 1) THEN
    IF MID$(ST$, I, 1) = MID$(ST$, I + 2, 1) THEN
      CORRECT = 0
    END IF
  END IF
  I = I + 1
WEND
IF CORRECT THEN PRINT "CORRECT" ELSE PRINT "MISPELLED"

```

'3.2

' This program finds the positive root of V for an equation.

,

```
DEF FNC (V) = -23511.9 * V * V + 988686.1 * V - 400943!
DEF FNB (V) = P(I) * V * 9062.599
DEF FNA (V) = P(I) * V * V * V * 14.14 - FNB(V) + FNC(V)
DATA 0.05, 0.7, 10.0, 70.0
FOR I = 1 TO 4: READ P(I): NEXT I
FOR I = 1 TO 5
  IF I = 5 THEN PRINT : INPUT "Enter value for P:"; P(5)
  FOR J = 0 TO 2
    IF SGN(FNA(J)) <> SGN(FNA(J + 1)) AND FNA(J + 1) <> 0 THEN
      LOW = J: HIGH = J + 1
      IF FNA(LOW) > FNA(HIGH) THEN SWAP LOW, HIGH
      WHILE ABS(LOW - HIGH) > .00005
        MID = (LOW + HIGH) / 2
        IF FNA(MID) < 0 THEN LOW = MID ELSE HIGH = MID
      WEND
      MID = SGN(MID) * INT(ABS(MID) * 10000 + .5) / 10000
      PRINT USING "P = ##.##"; P(I);
      PRINT USING "  V = #####"; MID
    END IF
  NEXT J
NEXT I
```

```

'3.3
' This program will magnify an input positive integer.
,
DATA 123567,36,13457,13467,2346,12467,124567,136,1234567,12346
FOR I = 0 TO 9: READ NUM$(I): NEXT I
INPUT "Enter number:"; N$
INPUT "Enter magnification:"; MAGN
CLS
FOR I = 1 TO LEN(N$)
  N = VAL(MID$(N$, I, 1))
  COL = (I - 1) * MAGN * 6 + 1
  FOR J = 1 TO LEN(NUM$(N))
    PART = VAL(MID$(NUM$(N), J, 1))
    GOSUB DisplayPart
  NEXT J
NEXT I
END
'
DisplayPart:
  SELECT CASE PART
    CASE 1
      LOCATE 1, COL
      FOR K = 1 TO MAGN: PRINT "*****"; : NEXT K: PRINT
    CASE 2
      FOR K = 1 TO MAGN * 2 + 1: LOCATE K, COL: PRINT "*": NEXT K
    CASE 3
      FOR K = 1 TO MAGN * 2 + 1
        LOCATE K, COL + MAGN * 4 - 1: PRINT "*"
      NEXT K
    CASE 4
      LOCATE MAGN * 2 + 1, COL
      FOR K = 1 TO MAGN: PRINT "*****"; : NEXT K: PRINT
    CASE 5
      FOR K = MAGN * 2 + 1 TO MAGN * 4 + 1
        LOCATE K, COL: PRINT "*"
      NEXT K
    CASE 6
      FOR K = MAGN * 2 + 1 TO MAGN * 4 + 1
        LOCATE K, COL + MAGN * 4 - 1: PRINT "*"
      NEXT K
    CASE 7
      LOCATE MAGN * 4 + 1, COL
      FOR K = 1 TO MAGN: PRINT "*****"; : NEXT K: PRINT
  END SELECT
RETURN

```

'3.4

' This program produces a calendar for a given month/year.

' January 1, 1901 is a Tuesday.

'

```

DIM MO$(12), DAYSINMO(12)
DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY
DATA AUGUST,SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER
DATA 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
FOR I = 1 TO 12: READ MO$(I): NEXT I
FOR I = 1 TO 12: READ DAYSINMO(I): NEXT I
INPUT "Enter month, year:"; MONTH, YEAR
MD = 2 + INT((26 - (LEN(MO$(MONTH)) + 5)) / 2)
CLS : PRINT SPACE$(MD); MO$(MONTH); YEAR
PRINT "  S  M  T  W  T  F  S"
PRINT "  -----"
' Calculate # of days from 1/1/1901 to last day of prior month
DAYS = (YEAR - 1901) * 365 + INT((YEAR - 1901) / 4)
FOR I = 1 TO MONTH - 1
  DAYS = DAYS + DAYSINMO(I)
NEXT I
IF (MONTH > 2) AND (YEAR MOD 4 = 0) THEN DAYS = DAYS + 1
' Determine first day of month
DAY = (DAYS + 1) MOD 7 'Day =0 (Mon), =1 (Tue) ... =6 (Sun)
COL = (DAY + 1) MOD 7 ' Day = 0,1,2,3,4,5,6 Sun,Mon...Sat
IF (MONTH = 2) AND (YEAR MOD 4 = 0) THEN LEAP = 1 ELSE LEAP = 0
' Display month calendar
IF COL > 0 THEN PRINT SPACE$(COL * 4);
FOR I = 1 TO DAYSINMO(MONTH) + LEAP
  PRINT USING "####"; I;
  COL = (COL + 1) MOD 7
  IF COL = 0 THEN PRINT
NEXT I

```

```
'3.5
' This program positions 5 queens on the board so none attack.
'
PRINT "ROWS = 1 2 3 4 5"
PRINT "-----"
PRINT "COLUMNS"
COL = 1: ROW = 1: DIMEN = 5
WHILE (COL > 1) OR (ROW < DIMEN + 1)
  WHILE (ROW <= DIMEN) AND (COL <= DIMEN)
    GOSUB IsQueenSafe
    IF SAFETY THEN
      CONFIG(COL) = ROW: COL = COL + 1: ROW = 1
    ELSE
      ROW = ROW + 1
    END IF
  WEND
  IF (ROW = DIMEN + 1) THEN COL = COL - 1: ROW = CONFIG(COL) + 1
  IF (COL = DIMEN + 1) THEN
'   Display solution and retreat column
    PRINT SPACE$(6);
    FOR I = 1 TO DIMEN: PRINT USING "##"; CONFIG(I); : NEXT I
    PRINT
    COL = COL - 1: ROW = CONFIG(COL) + 1
  END IF
WEND
END
' ----- Function Safety returns True if no queen can attack
IsQueenSafe:
  SAFETY = -1
  FOR I = 1 TO COL - 1
    IF (CONFIG(I) + I) = (ROW + COL) THEN SAFETY = 0
    IF (CONFIG(I) - I) = (ROW - COL) THEN SAFETY = 0
    IF (CONFIG(I) = ROW) THEN SAFETY = 0
  NEXT I
RETURN
```

'3.6
 ' This program prints the product of 2 large integers in Base.
 '

```

DEFINT A-Z
DIM A(31), B(31), PROD(61)
INPUT "Enter base:"; BAS
INPUT "Enter first integer: "; ASTR$
INPUT "Enter second integer:"; BSTR$
' -- Determine if signs are positive or negative
SIGN = 1
IF MID$(ASTR$, 1, 1) = "-" THEN
  ASTR$ = MID$(ASTR$, 2, LEN(ASTR$) - 1): SIGN = -1
END IF
IF MID$(BSTR$, 1, 1) = "-" THEN
  BSTR$ = MID$(BSTR$, 2, LEN(BSTR$) - 1): SIGN = SIGN * -1
END IF
IF SIGN < 0 THEN PRINT "-";
' -- Store string digits into numerical arrays
LENA = LEN(ASTR$): LENB = LEN(BSTR$)
FOR I = LENA TO 1 STEP -1
  A(LENA - I + 1) = VAL(MID$(ASTR$, I, 1))
NEXT I
FOR I = LENB TO 1 STEP -1
  B(LENB - I + 1) = VAL(MID$(BSTR$, I, 1))
NEXT I
' -- Multiply 2 numbers as a person would, with carries
FOR I = 1 TO LENB
  CARRY = 0
  FOR J = 1 TO LENA
    S = I + J - 1
    PROD(S) = PROD(S) + B(I) * A(J) + CARRY
    CARRY = INT(PROD(S) / BAS)
    PROD(S) = PROD(S) - CARRY * BAS
  NEXT J
  IF CARRY > 0 THEN PROD(S + 1) = CARRY
NEXT I
' -- Display product
IF CARRY > 0 THEN PRINT USING "#"; PROD(S + 1);
FOR I = S TO 1 STEP -1: PRINT USING "#"; PROD(I); : NEXT I

```

'3.7
 ' This program computes most efficient change without a coin.
 '

```

INPUT "Enter cost, amount:"; COST, AMOUNT
INPUT "Enter missing coin:"; COIN$
CHANGE = INT((AMOUNT - COST) * 100 + .1)
C$(1) = "QUARTER": C$(2) = "DIME": C$(3) = "NICKEL": C$(4) =
"PENNY"
A(1) = 25: A(2) = 10: A(3) = 5: A(4) = 1
X = CHANGE
ST = 1: EN = 4: GOSUB MakeChange 'Calculate denominations
C = 1

```

```

WHILE (C < 4) AND COIN$ <> C$(C): C = C + 1: WEND
SELECT CASE C
  CASE 1
    ' *** NO quarters ***
    ' Determine most efficient way without quarters (C=1)
    X = CHANGE
    ST = 2: EN = 4: GOSUB MakeChange 'Calculate denominations
  CASE 2
    ' *** NO dimes ***
    ' Add 2 nickels for every dime
    B(3) = B(3) + B(2) * 2
  CASE 3
    ' *** NO nickels ***
    ' IF a nickel then IF at least 1 quarter then
    ' Make 3 dimes and 1 less quarter
    ' Else make 5 more pennies with the 1 nickel
    IF B(3) = 1 THEN
      IF B(1) > 0 THEN
        B(2) = B(2) + 3: B(1) = B(1) - 1
      ELSE
        B(4) = B(4) + 5
      END IF
    END IF
END SELECT
'
' Display results
'
FOR I = 4 TO 1 STEP -1
  IF I <> C THEN
    PRINT USING "# "; B(I);
    IF I = 4 AND B(I) <> 1 THEN
      PRINT "PENNIES"
    ELSE
      PRINT C$(I); : IF B(I) <> 1 THEN PRINT "S" ELSE PRINT
    END IF
  END IF
NEXT I
PRINT "TOTAL CHANGE RETURNED ="; CHANGE; "CENT";
IF CHANGE <> 1 THEN PRINT "S" ELSE PRINT
END
'
' Determine most efficient change given coins
'
MakeChange:
  FOR I = ST TO EN
    B(I) = INT(X / A(I))
    X = X - B(I) * A(I)
  NEXT I
  RETURN

```

```

'3.8
' This program displays the coordinates of binary rectangles.
,
DEFINT A-Z
DIM A(6, 7)
' Convert 6 numbers to binary representation
FOR I = 1 TO 6
  INPUT "Enter number:"; NUM
  DEN = 128
  FOR J = 6 TO 0 STEP -1
    DEN = DEN / 2
    A(I, 7 - J) = INT(NUM / DEN)
    NUM = NUM - A(I, 7 - J) * DEN
  NEXT J
NEXT I
PRINT
' Display the 6 row X 7 col grid of 0s and 1s
FOR I = 1 TO 6
  FOR J = 1 TO 7
    PRINT USING "#"; A(I, J);
  NEXT J: PRINT
NEXT I
PRINT
' Find largest solid rectangles of 1s
FOR ROWLEN = 6 TO 2 STEP -1
  FOR COLLEN = 7 TO 2 STEP -1
    FOR ROWST = 1 TO 7 - ROWLEN
      FOR COLST = 1 TO 8 - COLLEN
        RECT = -1
        FOR I = ROWST TO ROWST + ROWLEN - 1
          J = COLST
          WHILE (J <= COLST + COLLEN - 1) AND RECT
            IF A(I, J) = 0 THEN RECT = 0
            J = J + 1
          WEND
        NEXT I
        IF RECT THEN
          PRINT USING "(#"; ROWST; : PRINT ", ";
          PRINT USING "#"; COLST; : PRINT ")";
          PRINT USING "(#"; ROWST + ROWLEN - 1; : PRINT ", ";
          PRINT USING "#"; COLST + COLLEN - 1; : PRINT ")";
          FOR I = ROWST TO ROWST + ROWLEN - 1
            FOR J = COLST TO COLST + COLLEN - 1
              A(I, J) = 0
            NEXT J
          NEXT I
        END IF
      NEXT COLST
    NEXT ROWST
  NEXT COLLEN
NEXT ROWLEN

```

```

'3.9
' This program determines the 5 word combination for BINGO.
'
DIM LETVAL(26)
DATA 9, 14, 1, 16, 20, 5, 10, 2, 21, 17, 6, 25
DATA 12, 3, 22, 18, 24, 7, 13, 26, 15, 11, 19, 4, 23, 8
FOR I = 1 TO 26: READ LETVAL(I): NEXT I
DATA BIBLE, IDYLL, NOISE, GULLY, OBESE
DATA OBESE, TITHE, INLET, IGLOO, TOWER
FOR COL = 1 TO 2
  FOR ROW = 1 TO 5
    READ HIGHWORDS$(ROW, COL): SUM = 0
    FOR I = 1 TO 5
      WORD$ = HIGHWORDS$(ROW, COL)
      LETTER$ = MID$(WORD$, I, 1)
      SUM = SUM + LETVAL(ASC(LETTER$) - 64)
    NEXT I
    HIGHEST(ROW, COL) = SUM
  NEXT ROW
NEXT COL
'
WHILE WORD$ <> "QUIT"
  GOSUB DisplayValues 'DisplayValues
  INPUT "Enter word: "; WORD$
  WHILE LEN(WORD$) = 5
    SUM = 0
    FOR I = 1 TO 5
      LETTER$ = MID$(WORD$, I, 1)
      LETTERS$(I) = LETTER$
      SUM = SUM + LETVAL(ASC(LETTER$) - 64)
    NEXT I
    GOSUB UseWord
    INPUT "Enter word: "; WORD$
  WEND
WEND
END
'
'-- Procedure UseWord
UseWord:
  FOR COL = 1 TO 2
    FOR ROW = 1 TO 5
      IF LETTERS$(COL) = MID$("BINGO", ROW, 1) THEN
        IF SUM > HIGHEST(ROW, COL) THEN
          HIGHEST(ROW, COL) = SUM: HIGHWORDS$(ROW, COL) = WORD$
        END IF
      END IF
    NEXT ROW
  NEXT COL
  RETURN
'
'-- Procedure DisplayValues
DisplayValues:
  PRINT : MAX = 0
  FOR I = 1 TO 2: MAXSUM(I) = 0: NEXT I

```

```

ST = 1: EN = 2
FOR ROW = 1 TO 5
  FOR COL = ST TO EN
    PRINT HIGHWORD$(ROW, COL);
    PRINT USING " ###"; HIGHEST(ROW, COL);
    PRINT SPACE$(3);
    MAXSUM(COL) = MAXSUM(COL) + HIGHEST(ROW, COL)
  NEXT COL
  PRINT
NEXT ROW
' Determine maximum column and display ***
FOR COL = ST TO EN
  PRINT SPACE$(3 + COL * 3); : PRINT USING "###"; MAXSUM(COL);
  IF MAXSUM(COL) > MAX THEN MAX = MAXSUM(COL): MAXCOL = COL
NEXT COL
PRINT
IF MAXCOL = 1 THEN
  PRINT SPACE$(6); "****"
ELSE
  PRINT SPACE$(18); "****"
END IF
PRINT
RETURN

```

'3.10

' This program displays the number of distinguishable
' permutations for a cube w/sides input as color symbols.
'

```

DIM UNIQUE$(24, 6)
DATA TOP,FRONT,BOTTOM,BACK,RIGHT,LEFT
FOR I = 1 TO 6: READ SIDE$(I): NEXT I
' Assign colors to original 4 cubes
FOR I = 1 TO 6
  PRINT "Enter "; SIDE$(I); " side:"; : INPUT CUBE$(I)
NEXT I
NUM = 0
' Rotate cubes and check if it is unique
FOR ROT = 0 TO 23
  GOSUB Permute
  IF ROT = 0 THEN
    VALID = -1
  ELSE
    J = 1: VALID = -1
    WHILE (J <= NUM) AND VALID
      VALID = 0
      FOR K = 1 TO 6
        IF C$(K) <> UNIQUE$(J, K) THEN VALID = -1
      NEXT K
      J = J + 1
    WEND
  END IF
  IF VALID THEN
    NUM = NUM + 1
  END IF

```

```
    FOR I = 1 TO 6: UNIQUE$(NUM, I) = C$(I): NEXT I
  END IF
NEXT ROT
PRINT "NUMBER OF DISTINGUISHABLE CUBES ="; NUM
END
'-- PROCEDURE THAT PERMUTES (SWAPS THE COLORS ON THE SQUARES)
Permute:
  IF ROT MOD 4 > 0 THEN
    TEMP$ = C$(2): C$(2) = C$(5): C$(5) = C$(4)
    C$(4) = C$(6): C$(6) = TEMP$
  ELSE
    SQUARE = INT(ROT / 4) + 1
    C$(1) = CUBE$(SQUARE)
    SELECT CASE SQUARE
      CASE 1
        FOR I = 2 TO 6: C$(I) = CUBE$(I): NEXT I
      CASE 2
        C$(2) = CUBE$(3): C$(3) = CUBE$(4)
        C$(4) = CUBE$(1): C$(5) = CUBE$(5): C$(6) = CUBE$(6)
      CASE 3
        C$(2) = CUBE$(4): C$(3) = CUBE$(1)
        C$(4) = CUBE$(2): C$(5) = CUBE$(5): C$(6) = CUBE$(6)
      CASE 4
        C$(2) = CUBE$(1): C$(3) = CUBE$(2)
        C$(4) = CUBE$(3): C$(5) = CUBE$(5): C$(6) = CUBE$(6)
      CASE 5
        C$(2) = CUBE$(2): C$(3) = CUBE$(6)
        C$(4) = CUBE$(4): C$(5) = CUBE$(3): C$(6) = CUBE$(1)
      CASE 6
        C$(2) = CUBE$(2): C$(3) = CUBE$(5)
        C$(4) = CUBE$(4): C$(5) = CUBE$(1): C$(6) = CUBE$(3)
    END SELECT
  END IF
RETURN
```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '90
BASIC PROGRAM SOLUTIONS

'1.1
' This program will display the initials of NCNB.
,

```
PRINT "NN      N  CCCCC  NN      N  BBBB"  
PRINT "N N    N  C          N N    N  B  B"  
PRINT "N  N  N  C          N  N  N  BBBBB"  
PRINT "N    N N  C          N    N N  B  B"  
PRINT "N      NN  CCCCC  N      NN  BBBB"
```

'1.2
' This program will print the name of the SYSTEM.
,

```
INPUT "Enter #:"; N  
PRINT "SYSTEM"; N
```

'1.3
' This program will display the value of programmers.
,

```
INPUT "Enter N: "; N  
PRINT 66 + N; "BILLION DOLLARS"
```

'1.4
' This program will indicate the county for zip code.
,

```
INPUT "Enter zip code: "; N  
IF N = 33701! OR N = 34685! OR N = 34646! THEN  
    PRINT "PINELLAS"  
ELSE  
    IF N = 33525! OR N = 34249! OR N = 34690! THEN  
        PRINT "PASCO"  
    ELSE  
        PRINT "HILLSBOROUGH"  
    END IF  
END IF
```

'1.5
' This program will display Hugh McColl's goals.
,

```
INPUT "Enter MMM: "; M  
INPUT "Enter YYYY: "; Y  
PRINT "HUGH MCCOLL WOULD LIKE NCNB TO GROW"  
PRINT "TO"; M; "BILLION DOLLARS IN ASSETS BY"  
PRINT "THE YEAR"; Y
```

'1.6

' This program will calculate maximum number of coupons.

,

INPUT "Enter N associates: "; N

INPUT "Enter C coupons: "; C

PRINT INT(C / N + .99)

'1.7

' This program will print divisions in COBOL program.

,

INPUT "Enter division: "; D\$

SELECT CASE D\$

 CASE "IDENTIFICATION"

 PRINT "BEFORE = NONE"

 PRINT "AFTER = ENVIRONMENT DATA PROCEDURE"

 CASE "ENVIRONMENT"

 PRINT "BEFORE = IDENTIFICATION"

 PRINT "AFTER = DATA PROCEDURE"

 CASE "DATA"

 PRINT "BEFORE = IDENTIFICATION ENVIRONMENT"

 PRINT "AFTER = PROCEDURE"

 CASE "PROCEDURE"

 PRINT "BEFORE = IDENTIFICATION DATA PROCEDURE"

 PRINT "AFTER = NONE"

END SELECT

'1.8

' This program will display states having holidays.

,

INPUT "Enter N: "; N

IF N <= 7 THEN PRINT "FL NC SC TX MD GA VA": END

IF N = 8 THEN PRINT "FL NC TX MD GA VA": END

IF N = 9 OR N = 10 THEN PRINT "FL TX MD GA VA": END

IF N = 11 THEN PRINT "MD"

'1.9

' This program will correct modern dates.

,

INPUT "Enter date: "; D

INPUT "Enter A.D. or B.C.:"; A\$

IF A\$ = "B.C." AND D > 4 THEN PRINT D - 4; "B.C.": END

IF A\$ = "B.C." THEN PRINT 5 - D; "A.D.": END

PRINT D + 4; "A.D"

'1.10

' This program will print a 7 letter word diamond.

,

```
INPUT "Enter word: "; N$
PRINT " "; MID$(N$, 4, 1)
PRINT " "; MID$(N$, 3, 3)
PRINT " "; MID$(N$, 2, 5)
PRINT MID$(N$, 1, 7)
PRINT " "; MID$(N$, 2, 5)
PRINT " "; MID$(N$, 3, 3)
PRINT " "; MID$(N$, 4, 1)
```

'2.1

' This program will encode a phrase.

,

```
INPUT "Enter phrase: "; A$
FOR I = 1 TO LEN(A$)
  C$ = MID$(A$, I, 1)
  IF C$ < "A" OR C$ > "Z" THEN
    PRINT C$;
  ELSE
    IF C$ = "A" THEN
      PRINT "Z";
    ELSE
      PRINT CHR$(ASC(C$) - 1);
    END IF
  END IF
END IF
NEXT I
```

'2.2

' This program will determine the "type" of year.

,

```
INPUT "Enter year: "; Y
IF Y / 10 = INT(Y / 10) THEN PRINT "END OF DECADE"
IF Y / 100 = INT(Y / 100) THEN PRINT "END OF CENTURY"
IF Y / 1000 = INT(Y / 1000) THEN PRINT "END OF MILLENNIUM"
IF Y - INT(Y / 10) * 10 = 1 THEN PRINT "BEGINNING OF DECADE"
IF Y - INT(Y / 100) * 100 = 1 THEN PRINT "BEGINNING OF CENTURY"
IF Y - INT(Y / 1000) * 1000 = 1 THEN
  PRINT "BEGINNING OF MILLENNIUM"
END IF
```

'2.3

' This program will print average and handicap of bowlers.

,

```
A$(1) = "BOB:   "; A$(2) = "DOUG:   "; A$(3) = "JACKIE:"
A$(4) = "JOSE:  "
FOR I = 1 TO 4
  PRINT "Enter scores for "; A$(I); : INPUT S1, S2, S3
  AVE(I) = (S1 + S2 + S3) / 3
  IF AVE(I) > 200 THEN
    HAN(I) = 0
  ELSE
    HAN(I) = (200 - AVE(I)) * .9
  END IF
NEXT I
FOR I = 1 TO 4
  PRINT A$(I);
  PRINT USING " AVERAGE = ###"; INT(AVE(I) + .01);
  PRINT "  HANDICAP ="; INT(HAN(I) + .01)
NEXT I
```

'2.4

' This program will determine # of days to add to date.

,

```
INPUT "Enter date: "; D$
MM = VAL(LEFT$(D$, 2))
DD = VAL(MID$(D$, 4, 2))
YY = VAL(RIGHT$(D$, 4))
PRINT "ADD ";
IF YY < 1700 OR (YY = 1700 AND MM < 3) THEN PRINT "10 DAYS": END
IF YY < 1800 OR (YY = 1800 AND MM < 3) THEN PRINT "11 DAYS": END
IF YY < 1900 OR (YY = 1900 AND MM < 3) THEN PRINT "12 DAYS": END
IF YY < 2100 OR (YY = 2100 AND MM < 3) THEN PRINT "13 DAYS": END
```

'2.5

' This program will sort efficiencies of sorting algorithms.

,

```
N$(1) = "BUBBLE SORT": N$(2) = "SHELL SORT": N$(3) = "QUICK SORT"
INPUT "Enter N: "; N
A(1) = N * (N - 1) / 2
A(2) = N * (LOG(N) / LOG(2)) * (LOG(N) / LOG(2))
A(3) = N * (LOG(N) / LOG(2))
FOR I = 1 TO 2
  FOR J = I + 1 TO 3
    IF A(I) > A(J) THEN
      SWAP A(I), A(J): SWAP N$(I), N$(J)
    END IF
  NEXT J
NEXT I
FOR I = 1 TO 3: PRINT N$(I): NEXT I
```

'2.6

' This program will determine status for each hole of golf.

,

```
DATA 4, 3, 4, 5, 4, 3, 5, 4, 4
FOR I = 1 TO 9: READ P(I): PAR = PAR + P(I): NEXT I
FOR I = 1 TO 9
  PRINT "Enter score for hole"; I; : INPUT S(I)
  SUM = SUM + S(I)
NEXT I
PRINT "HOLE  PAR  SCORE  STATUS"
PRINT "----  ---  -----  -----"
FOR I = 1 TO 9
  PRINT I; "  "; P(I); "  "; S(I); "  ";
  D = S(I) - P(I)
  SELECT CASE D
    CASE -3: PRINT "DOUBLE EAGLE"
    CASE -2: PRINT "EAGLE"
    CASE -1: PRINT "BIRDIE"
    CASE 0: PRINT "PAR"
    CASE 1: PRINT "BOGEY"
    CASE 2: PRINT "DOUBLE BOGEY"
  END SELECT
NEXT I
PRINT "  ---  -----"
PRINT "  "; PAR; "  "; SUM
```

'2.7

' This program will determine time calendar is ahead/behind.

,

```
INPUT "Enter N: "; N
' Sum 5 hours 48 min 47.8 sec for every year
H = 5 * N: M = 48 * N: S = 47.8 * N
' Convert to standard form
SN = INT(S / 60): S = S - SN * 60: M = M + SN
MN = INT(M / 60): M = M - MN * 60: H = H + MN
HN = INT(H / 24): H = H - HN * 24: D = HN
' Subtract 1 for every leap year counted
LY = INT(N / 4)
IF LY <= D THEN
  PRINT D - LY; "DAYS "; H; "HOURS "; M; "MIN ";
  PRINT USING "##.# SEC AHEAD"; S
ELSE
  PRINT (LY - D - 1); "DAYS ";
  PRINT 23 - H; "HOURS "; 59 - M; "MIN ";
  PRINT USING "##.# SEC BEHIND"; 60 - S
END IF
```

'2.8

' This program will display members on a committee.

```

,
DATA JACKIE,TOM,LOVETTA,GREG,TONY,AL,KAREN
DATA JAN,NORM,TRUDY,THERESA,ALICE,DAVE,JIM,STEVE
DIM A$(20)
FOR I = 1 TO 15: READ A$(I): NEXT I
N$(1) = "BARB": NM(1) = 6: N$(2) = "JOE": NM(2) = 8
N$(3) = "DOUG": NM(3) = 9: Y = 1989: M = 9
INPUT "Enter month, year: "; MONTH, YEAR
PRINT USING "##/"; M; : PRINT USING "#### - "; Y;
PRINT N$(1); " "; N$(2); " "; N$(3)
I = 1
WHILE (M <> MONTH) OR (Y <> YEAR)
  M = M + 1: IF M = 13 THEN M = 1: Y = Y + 1
  FOR J = 1 TO 3
    IF ABS(M - NM(J)) = 6 THEN
      N$(J) = A$(I): I = I + 1: NM(J) = M
      PRINT USING "##/"; M; : PRINT USING "#### - "; Y;
      PRINT N$(1); " "; N$(2); " "; N$(3)
    END IF
  NEXT J
WEND

```

'2.9

' This program will graph the sine and cosine functions.

```

,
FOR F = 1 TO 2
  CLS
  FOR I = 1 TO 24: LOCATE I, 40: PRINT "!"; : NEXT I
  LOCATE 12, 1
  FOR I = 1 TO 79: PRINT "-"; : NEXT I
  LOCATE 12, 40: PRINT "+";
  CINC = 39 / 3.14: RINC = 11
  FOR X = -3.14 TO 3.14 STEP .05
    C = 40 + CINC * X
    IF F = 1 THEN R = 12 - SIN(X) * RINC
    IF F = 2 THEN R = 12 - COS(X) * RINC
    LOCATE R, C: PRINT "*";
  NEXT X
  A$ = "": WHILE A$ = "": A$ = INKEY$: WEND
NEXT F
CLS

```

```

'2.10
' This program will estimate hours of training given choices.
'
CLS
PRINT "          NCNB IN-HOUSE TRAINING LIST"
PRINT
PRINT "COURSE #      COURSE NAME                      EST. HOURS"
PRINT "-----"
A$(1) = "187-11X": B$(1) = "ISPF/PDS FUNDAMENTALS      6.5 - 8"
A$(2) = "187-15X": B$(2) = "ISPF/PDS FOR PROGRAMMERS  4.5 - 6"
A$(3) = "220-AXX": B$(3) = "JCL FUNDAMENTALS          15 - 20"
A$(4) = "200-AXX": B$(4) = "VSAM CONCEPTS           4 - 7"
A$(5) = "123-2XX": B$(5) = "MVS/SP/XA VSAM           7 - 11"
A$(6) = "130-11X": B$(6) = "CICS/VS SKILLS I           6 - 8"
A$(7) = "130-15X": B$(7) = "CICS/VS SKILLS II          4 - 6"
DATA 6.5,8, 4.5,6, 15,20, 4,7, 7,11, 6,8, 4,6
FOR I = 1 TO 7: READ LOW(I), HIGH(I): NEXT I
FOR I = 1 TO 7
  PRINT A$(I); "      "; B$(I)
NEXT I
PRINT : NUM = 0
INPUT "Enter course # (or 000-000 to end): "; C$
WHILE C$ <> "000-000"
  I = 1: WHILE C$ <> A$(I): I = I + 1: WEND
  NUM = NUM + 1: C(NUM) = I
  LSUM = LSUM + LOW(I): HSUM = HSUM + HIGH(I)
  INPUT "Enter course # (or 000-000 to end): "; C$
WEND
'   Display options selected and TOTAL estimated hours
CLS
PRINT "COURSE NAME                      EST. HOURS"
PRINT "-----"
FOR I = 1 TO NUM: PRINT B$(C(I)): NEXT I
PRINT "
PRINT USING "          TOTAL = ##.##"; LSUM;
PRINT " -"; HSUM; "HOURS"

```

```

'3.1
' This program will produce acronyms for phone numbers.
,
DIM A$(18)
DATA AGENT, SOAP, MONEY, JEWEL, BALL, LOANS, CARE, SAVE, CALL
DATA PAVE, KEEP, KINGS, KNIFE, KNOCK, JOINT, JUICE, LOBBY, RATE
FOR I = 1 TO 18: READ A$(I): NEXT I
DATA A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, R, S, T, U, V, W, X, Y
FOR I = 2 TO 9: READ L1$(I), L2$(I), L3$(I): NEXT I
INPUT "Enter phone #: "; PH$
P4$ = RIGHT$(PH$, 4): P5$ = MID$(PH$, 3, 1) + P4$
' Convert words to number strings
FOR I = 1 TO 18
  L = LEN(A$(I)): NUM$ = ""
  FOR J = 1 TO L
    K = 2: C$ = MID$(A$(I), J, 1)
    WHILE (L1$(K) <> C$) AND (L2$(K) <> C$) AND (L3$(K) <> C$)
      K = K + 1
    WEND
    NUM$ = NUM$ + LTRIM$(STR$(K))
  NEXT J
  IF L = 4 AND NUM$ = P4$ THEN
    PRINT LEFT$(PH$, 4); A$(I)
  ELSE
    IF L = 5 AND NUM$ = P5$ THEN
      PRINT LEFT$(PH$, 2); LEFT$(A$(I), 1); "-";
      PRINT RIGHT$(A$(I), 4)
    END IF
  END IF
NEXT I

```

'3.2

' This program will select words given a string w/ wildcard.

,

```
DATA COMPUTE, COMPUTER, COMPUTERS, COMPORT, COMPUTES
DATA COMPUTED, ATTRACTIVE, ABRASIVE, ADAPTIVE, ACCEPTIVE
DATA AERATING, CONTESTED, CONTESTER, CORONETS, CONTESTS
DATA CONTESTERS, COUNTESS, CREATIVE, CREATE, CREATURE
DATA CREATION, EVERYBODY, EVERYONE, EMPTY, ELECTION
DIM A$(25)
N = 25: FOR I = 1 TO N: READ A$(I): NEXT I: I = 0
DO UNTIL I > L
  INPUT "Enter string: "; A$: L = LEN(A$): W = 0: I = 0: X$ = ""
  WHILE (I <= L) AND (X$ <> "*")
    I = I + 1: X$ = MID$(A$, I, 1)
  WEND
  IF I > L THEN END
  Asterisk is position I
  L$ = LEFT$(A$, I - 1): R$ = RIGHT$(A$, L - I)
  FOR J = 1 TO N
    IF LEFT$(A$(J), I - 1) = L$ AND RIGHT$(A$(J), L - I) = R$ THEN
      PRINT A$(J); " "; : W = 1
    END IF
  NEXT J
  IF W = 0 THEN PRINT "NO WORDS FOUND"
  PRINT
LOOP
```

```

'3.3
' This program will keep score for a double dual race.
'
CLS : DIM IN$(21)
FOR I = 1 TO 21
  PRINT "Place "; I; ":"; : INPUT IN$(I)
  IF I > 1 THEN
    J = 1
    WHILE J <= TN AND INIT$(J) <> IN$(I): J = J + 1: WEND
  END IF
  IF (INIT$(J) <> IN$(I)) OR (I = 1) THEN
    TN = TN + 1: INIT$(TN) = IN$(I)
  END IF
NEXT I
' Assert TEAM$(1, 2, 3) = 3 unique team INITIALS
FOR I = 1 TO 2
  FOR J = I + 1 TO 3
    PL = 0: T1 = 0: T2 = 0: T1PL = 0: T2PL = 0
    FOR K = 1 TO 21
      IF IN$(K) = INIT$(I) THEN
        PL = PL + 1: T1 = T1 + PL: T1PL = T1PL + 1
        TEAM1(T1PL) = PL
      END IF
      IF IN$(K) = INIT$(J) THEN
        PL = PL + 1: T2 = T2 + PL: T2PL = T2PL + 1
        TEAM2(T2PL) = PL
      END IF
    NEXT K
    T1 = T1 - TEAM1(6) - TEAM1(7)
    T2 = T2 - TEAM2(6) - TEAM2(7)
    PRINT "TEAM "; INIT$(I); ":"; T1; " POINTS"
    PRINT "TEAM "; INIT$(J); ":"; T2; " POINTS"
    IF (T1 < T2) OR (T1 = T2 AND TEAM1(6) < TEAM2(6)) THEN
      PRINT "TEAM "; INIT$(I);
    ELSE
      PRINT "TEAM "; INIT$(J);
    END IF
    PRINT " WINS!": PRINT
  NEXT J
NEXT I

```

'3.4

' This program will determine who gets which program #s.

,

DEFINT A-Z

INPUT "Enter X, Y, Z: "; X, Y, Z

A\$(1) = "AL, DOUG, AND JAN = "

A\$(2) = "AL AND DOUG = "

A\$(3) = "AL AND JAN = "

A\$(4) = "DOUG AND JAN = "

A\$(5) = "AL = "

A\$(6) = "DOUG = "

A\$(7) = "JAN = "

A\$(8) = "NORM = "

FOR K = 1 TO 8

PRINT A\$(K); : ONE = 0

FOR I = 1 TO 30

XD = (I / X = INT(I / X)): YD = (I / Y = INT(I / Y))

ZD = (I / Z = INT(I / Z))

I\$ = LTRIM\$(STR\$(I)) + " "

IF K = 1 AND XD AND YD AND ZD THEN PRINT I\$; : ONE = 1

IF K = 2 AND XD AND YD AND NOT ZD THEN PRINT I\$; : ONE = 1

IF K = 3 AND XD AND NOT YD AND ZD THEN PRINT I\$; : ONE = 1

IF K = 4 AND NOT XD AND YD AND ZD THEN PRINT I\$; : ONE = 1

IF K = 5 AND XD AND NOT YD AND NOT ZD THEN PRINT I\$; : ONE = 1

IF K = 6 AND NOT XD AND YD AND NOT ZD THEN PRINT I\$; : ONE = 1

IF K = 7 AND NOT XD AND NOT YD AND ZD THEN PRINT I\$; : ONE = 1

IF K = 8 AND NOT XD AND NOT YD AND NOT ZD THEN

PRINT I\$; : ONE = 1

END IF

NEXT I

IF ONE = 0 THEN PRINT "NONE" ELSE PRINT

NEXT K

```

'3.5
' This program will display numbers 1-8 and a blank in a
' 3 x 3 array.  When a digit is pressed, it moves into the
' blank (if possible).
'
RANDOMIZE TIMER
' Assign numbers in array sequentially then scramble them
FOR I = 1 TO 3
  FOR J = 1 TO 3
    A(I, J) = (I - 1) * 3 + J - 1
  NEXT J
NEXT I
FOR I = 1 TO 3
  FOR J = 1 TO 3
    R1 = INT(RND(3) * 3) + 1: R2 = INT(RND(3) * 3) + 1
    X = A(I, J): A(I, J) = A(R1, R2): A(R1, R2) = X
  NEXT J
NEXT I
'
WHILE (DIG <> 9)
'   Display Array
  CLS
  FOR I = 1 TO 3
    FOR J = 1 TO 3
      IF A(I, J) > 0 THEN PRINT A(I, J); " ";
      IF A(I, J) = 0 THEN PRINT "    "; : BX = I: BY = J
    NEXT J: PRINT
  NEXT I
'   Accept valid digit or 9 (to end)
  VALID = 0
  WHILE (VALID = 0) AND (DIG <> 9)
    A$ = "": WHILE A$ = "": A$ = INKEY$: WEND
    DIG = VAL(A$)
    FOR I = 1 TO 3
      FOR J = 1 TO 3
        IF DIG = A(I, J) THEN IX = I: IY = J
      NEXT J
    NEXT I
    IF ABS(BX - IX) + ABS(BY - IY) = 1 THEN VALID = -1
  WEND
'
  IF VALID THEN
'   Move digit into blank space
    X = A(IX, IY): A(IX, IY) = A(BX, BY): A(BX, BY) = X
  END IF
WEND

```

'3.6

' This program will simulate the moves of a chess game.

```

'
A$(8) = "BR1 BK1 BB1 BQ BK BB2 BK2 BR2 ! 8"
A$(7) = "BP1 BP2 BP3 BP4 BP5 BP6 BP7 BP8 ! 7"
A$(6) = " ! 6"
A$(5) = " ! 5"
A$(4) = " ! 4"
A$(3) = " ! 3"
A$(2) = "WP1 WP2 WP3 WP4 WP5 WP6 WP7 WP8 ! 2"
A$(1) = "WR1 WK1 WB1 WQ WK WB2 WK2 WR2 ! 1"
A$(9) = "-----"
A$(10) = " A B C D E F G H"
CLS : L = LEN(A$(1))
FOR I = 8 TO 1 STEP -1: PRINT A$(I): NEXT I
PRINT A$(9): PRINT A$(10)
WKR = 1: WKC = 5: BKR = 8: BKC = 5 'Location of 2 kings
'
WHILE (R2 <> WKR OR C2 <> WKC) AND (R2 <> BKR OR C2 <> BKC)
  LOCATE 12, 1: PRINT SPACE$(30): LOCATE 12, 1
  IF MOV = 0 THEN INPUT "Enter white move: "; M$
  IF MOV = 1 THEN INPUT "Enter black move: "; M$
  ' Convert moves to coordinates
  C1 = ASC(LEFT$(M$, 1)) - 64: R1 = VAL(MID$(M$, 2, 1))
  C2 = ASC(MID$(M$, 4, 1)) - 64: R2 = VAL(RIGHT$(M$, 1))
  ' Move piece from 1 string to another and redisplay
  PIEC$ = MID$(A$(R1), (C1 - 1) * 4 + 1, 4)
  L$ = LEFT$(A$(R2), (C2 - 1) * 4)
  R$ = RIGHT$(A$(R2), L - C2 * 4)
  A$(R2) = L$ + PIEC$ + R$
  LOCATE 9 - R2, 1: PRINT A$(R2)
  ' Remove piece from string by placing spaces and redisplay
  L$ = LEFT$(A$(R1), (C1 - 1) * 4)
  R$ = RIGHT$(A$(R1), L - C1 * 4)
  A$(R1) = L$ + " " + R$
  LOCATE 9 - R1, 1: PRINT A$(R1)
  ' If a king moved, store new location
  IF R1 = WKR AND C1 = WKC THEN WKR = R2: WKC = C2: R2 = 0: C2 = 0
  IF R1 = BKR AND C1 = BKC THEN BKR = R2: BKC = C2: R2 = 0: C2 = 0
  IF MOV = 0 THEN MOV = 1 ELSE MOV = 0
WEND
LOCATE 12, 1: PRINT "CHECK MATE, ";
IF R2 = WKR AND C2 = WKC THEN PRINT "BLACK WON ": END
PRINT "WHITE WON "
```

```

'3.7
' This program will print date of Easter and Lent in a year.
,
DATA 4,14, 4,3, 3,23, 4,11, 3,31, 4,18, 4,8, 3,28, 4,16, 4,5
DATA 3,25, 4,13, 4,2, 3,22, 4,10, 3,30, 4,17, 4,7, 3,27
DIM M(18), D(18)
FOR I = 0 TO 18: READ M(I), D(I): NEXT I
MD(1) = 31: MD(2) = 28: MD(3) = 31
MO$(2) = "FEBRUARY": MO$(3) = "MARCH": MO$(4) = "APRIL"
INPUT "Enter year: "; Y
KE = Y - INT(Y / 19) * 19
' Calculate # of days between 1,1,1970 and date
DAYS = (Y - 1970) * 365 + INT((Y - 1968) / 4)
FOR I = 1 TO M(KE) - 1: DAYS = DAYS + MD(I): NEXT I
DAYS = DAYS + D(KE)
X = DAYS - INT(DAYS / 7) * 7
' if X = 0-Wed, 1-Thu, 2-Fri, 3-Sat, 4-Sun, 5-Mon, 6-Tue
IF X = 0 OR X = 1 OR X = 2 OR X = 3 THEN EDAY = D(KE) + (4 - X)
IF X = 4 OR X = 5 OR X = 6 THEN EDAY = D(KE) + (11 - X)
EMON = M(KE)
IF M(KE) = 3 AND EDAY > MD(3) THEN
  EDAY = EDAY - MD(3): EMON = EMON + 1
END IF
PRINT "EASTER IS ON "; MO$(EMON); EDAY
' Compute date of Lent
LMON = EMON - 1: LDAY = MD(LMON) + EDAY - 46
IF LDAY < 1 THEN LMON = LMON - 1: LDAY = LDAY + MD(LMON)
IF LMON = 2 AND Y / 4 = INT(Y / 4) THEN LDAY = LDAY + 1
PRINT "LENT IS ON "; MO$(LMON); LDAY

```

```

'3.8
' This program will keep score for a bowler.
'
DIM A(10, 3): WIDTH 40: CLS
FOR I = 1 TO 10: PRINT "Enter frame"; I; : INPUT A$(I): NEXT I
PRINT
PRINT "-1- -2- -3- -4- -5- -6- -7- -8- -9- -10-";
PRINT "----!----!----!----!----!----!----!----!----!----!";
FOR I = 1 TO 10
  PRINT SPACE$(3 - LEN(A$(I))); A$(I); "!";
NEXT I
'
' Assign values to A Frames according to X, /, or pins
'
FOR FR = 1 TO 10
  L = LEN(A$(FR))
  FOR J = 1 TO L
    MD$ = MID$(A$(FR), J, 1)
    IF MD$ = "X" THEN
      A(FR, J) = 10: LOOK(FR) = 2
    ELSE
      IF MD$ = "/" THEN
        A(FR, J) = 10 - A(FR, J - 1): LOOK(FR) = 1
      ELSE
        A(FR, J) = VAL(MD$)
      END IF
    END IF
  NEXT J
NEXT FR
'
' Determine FFrame values with LOOK ahead
'
FOR FR = 1 TO 10
  SUM(FR) = SUM(FR - 1) + A(FR, 1) + A(FR, 2)
  IF LOOK(FR) > 0 THEN
    IF LOOK(FR) <= 1 THEN
      ' *** A spare / needs 1 more value added ***
      IF FR = 10 THEN
        SUM(FR) = SUM(FR) + A(FR, 3)
      ELSE
        SUM(FR) = SUM(FR) + A(FR + 1, 1)
      END IF
    ELSE
      ' *** A strike X needs 2 more values added ***
      IF FR = 10 THEN
        SUM(FR) = SUM(FR) + A(FR, 3)
      ELSE
        SUM(FR) = SUM(FR) + A(FR + 1, 1) + A(FR + 1, 2)
        IF FR <> 9 THEN
          IF A(FR + 1, 1) = 10 THEN
            SUM(FR) = SUM(FR) + A(FR + 2, 1)
          END IF
        END IF
      END IF
    END IF
  END IF
END IF

```

```

END IF
'   *** Print FFrame's value ***
SUM$ = MID$(STR$(SUM(FR)), 2)
PRINT SUM$; SPACE$(3 - LEN(SUM$)); "!";
NEXT FR
PRINT STRING$(40, "-")

```

'3.9

' This program will solve an N x N system of equations.

```

'
INPUT "Enter N: "; N
FOR ROW = 1 TO N
  PRINT "Enter coefficients for row"; ROW
  FOR COL = 1 TO N
    PRINT "CO"; COL; ": "; : INPUT C(ROW, COL)
  NEXT COL
  INPUT "Enter constant: "; C(ROW, N + 1)
NEXT ROW
'   Make main diagonals all 1s with 0s to the left
FOR ROW = 1 TO N
  DEN = C(ROW, ROW)
  FOR COL = ROW TO N + 1
    C(ROW, COL) = C(ROW, COL) / DEN
  NEXT COL
  FOR R = ROW + 1 TO N
    X = C(R, ROW)
    FOR COL = ROW TO N + 1
      C(R, COL) = C(R, COL) - X * C(ROW, COL)
    NEXT COL
  NEXT R
NEXT ROW
'   Make 0s on the right of 1s on main diagonal, not const
FOR ROW = N TO 1 STEP -1
  FOR R = ROW - 1 TO 1 STEP -1
    X = C(R, ROW)
    FOR COL = ROW TO N + 1
      C(R, COL) = C(R, COL) - X * C(ROW, COL)
    NEXT COL
  NEXT R
NEXT ROW
'   Display solution
PRINT "("; LTRIM$(STR$(INT(C(1, N + 1) + .1)));
FOR ROW = 2 TO N: PRINT ", ";
  PRINT LTRIM$(STR$(INT(C(ROW, N + 1) + .1)));
NEXT ROW
PRINT ")"

```

```

'3.10
' This program will solve cryptarithms with two 2-letter addends
' and a 3-letter sum, using only the letters A, B, C, D, and E.
'
DEFINT A-Z
INPUT "Enter first addend: "; S1$
INPUT "Enter second addend: "; S2$
INPUT "Enter sum: "; S3$
L$ = S1$ + S2$ + S3$
' Store in FL() the index of the first occurrence
FOR I = 1 TO 7
  CH$ = MID$(L$, I, 1)
  J = 1: WHILE MID$(L$, J, 1) <> CH$: J = J + 1: WEND
  FL(I) = J
  IF J = I THEN NL = NL + 1: UL(NL) = I 'A new letter
NEXT I
'
FOR N1 = 10 TO 98 'N1 must be 2 digits, >9
  FOR N2 = 100 - N1 TO 98 'N2 must be 2 digits, >9
    SUM = N1 + N2 'Sum must be 3 digits >99
    N1$ = LTRIM$(STR$(N1))
    N2$ = LTRIM$(STR$(N2))
    SUM$ = LTRIM$(STR$(SUM))
    NS$ = N1$ + N2$ + SUM$
    I = 1: SOL = 1
    ' Check if similar letters correspond to similar #s
    WHILE (I <= 7) AND (SOL = 1)
      CH$ = MID$(NS$, I, 1)
      IF CH$ <> MID$(NS$, FL(I), 1) THEN SOL = 0
      I = I + 1
    WEND
    ' Check if unique letters correspond to unique digits
    FOR I = 1 TO NL - 1
      FOR J = I + 1 TO NL
        C1$ = MID$(NS$, UL(I), 1)
        C2$ = MID$(NS$, UL(J), 1)
        IF C1$ = C2$ THEN SOL = 0
      NEXT J
    NEXT I
    ' Display Solution
    IF SOL > 0 THEN
      FOR I = 1 TO NL
        PRINT MID$(L$, UL(I), 1); " = "; MID$(NS$, UL(I), 1)
      NEXT I
      PRINT : TOT = TOT + 1: END ' Only one needed
    END IF
  NEXT N2
NEXT N1
IF TOT = 0 THEN PRINT "NO SOLUTION POSSIBLE"

```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '91
BASIC PROGRAM SOLUTIONS

```
'1.1
' This program will display a phrase as a rectangle.
,
A$ = "COMPUTER CONTEST 1991"
CLS
PRINT A$: L = LEN(A$)
FOR I = 2 TO L - 1
  LOCATE I, 1: PRINT MID$(A$, I, 1)
  LOCATE I, L: PRINT MID$(A$, L - I + 1, 1)
NEXT I
FOR I = L TO 1 STEP -1
  PRINT MID$(A$, I, 1);
NEXT I

'1.2
' This program will display 2 random #s and their sum.
,
RANDOMIZE TIMER
X = INT(RND(3) * 19) - 9
Y = INT(RND(3) * 19) - 9
PRINT X; " + "; Y; " = "; X + Y

'1.3
' This program prints the total point score for a team.
,
INPUT "Enter team name: "; N$
INPUT "Enter # of 1 point programs: "; P1
INPUT "Enter # of 2 point programs: "; P2
INPUT "Enter # of 3 point programs: "; P3
TOT = P1 + P2 * 2 + P3 * 3
PRINT N$; " SCORED"; TOT; "POINTS"

'1.4
' This program displays a spreadsheet.
,
CLS
PRINT "  A B C D E F G H I J K L M N O P Q R S T"
FOR I = 1 TO 20: PRINT USING "##"; I: NEXT I

'1.5
' This program determines the number of teams competing.
,
INPUT "Enter number of students: "; X
PRINT X / 4; "TEAMS"
```

'1.6
' This program displays a word twice intersecting at a letter.
,

```
INPUT "Enter word: "; A$
INPUT "Enter letter: "; L$
X = INSTR(A$, L$)
CLS : LOCATE X, 1: PRINT A$
FOR I = 1 TO LEN(A$)
  LOCATE I, X: PRINT MID$(A$, I, 1)
NEXT I
```

'1.7
' This program displays fields from an account key.
,

```
INPUT "Enter account key: "; A$
PRINT "ORGANIZATION "; MID$(A$, 1, 3)
PRINT "BRANCH "; MID$(A$, 4, 3)
PRINT "DEALER "; MID$(A$, 7, 4)
PRINT "CLASS "; MID$(A$, 11, 3)
PRINT "UNIT "; MID$(A$, 14, 6)
```

'1.8
' This program displays the # of job steps in JCL.
,

```
INPUT "Enter line: "; L$
WHILE L$ <> "/"
  IF L$ = "EXEC" THEN ST = ST + 1
  INPUT "Enter line: "; L$
WEND
PRINT ST; "JOB STEPS"
```

'1.9
' This program will replace MAN with PERSON.
,

```
INPUT "Enter sentence: "; S$
FOR I = 1 TO LEN(S$)
  M$ = MID$(S$, I, 3)
  IF M$ = "MAN" THEN
    PRINT "PERSON"; : I = I + 2
  ELSE
    IF M$ = "MEN" THEN
      PRINT "PERSONS"; : I = I + 2
    ELSE
      PRINT MID$(S$, I, 1);
    END IF
  END IF
NEXT I
```

```
'1.10
' This program determines the winner of two computer teams.
'
INPUT "Enter team name: "; N1$
INPUT "Enter points, time, penalties: "; P1, T1, PEN1
'
INPUT "Enter team name: "; N2$
INPUT "Enter points, time, penalties: "; P2, T2, PEN2
'
IF P1 > P2 THEN W$ = N1$ ELSE IF P2 > P1 THEN W$ = N2$
H1 = INT(T1 / 100): M1 = T1 - H1 * 100
H2 = INT(T2 / 100): M2 = T2 - H2 * 100
TI1 = H1 * 60 + M1 + PEN1 * 5
TI2 = H2 * 60 + M2 + PEN2 * 5
IF P1 = P2 THEN IF TI1 < TI2 THEN W$ = N1$ ELSE W$ = N2$
PRINT W$; " WINS"
```

'2.1

' This program displays a pyramid of consecutive numbers.

,

```
INPUT "Enter N: "; N
```

```
S = 1
```

```
WHILE S < N
```

```
  I = I + 1
```

```
  PRINT SPACE$(20 - I * 2);
```

```
  FOR J = 1 TO I
```

```
    PRINT MID$(STR$(100 + S), 3, 2); "  ";
```

```
    S = S + 1
```

```
  NEXT J
```

```
  PRINT
```

```
WEND
```

'2.2

' This program will line up numbers with decimal points.

,

```
FOR I = 1 TO 5
```

```
  INPUT "Enter #: "; A$(I)
```

```
NEXT I
```

```
FOR I = 1 TO 5
```

```
  X = INSTR(A$(I), ".")
```

```
  PRINT SPACE$(6 - X); A$(I)
```

```
  SUM# = SUM# + VAL(A$(I))
```

```
NEXT I
```

```
PRINT " -----"
```

```
TOT$ = STR$(SUM# + .00001)           'Round off machine error
```

```
X = INSTR(TOT$, ".")
```

```
PRINT SPACE$(6 - X); MID$(TOT$, 1, X + 4) 'Round off error
```

'2.3

' This program will convert BASIC to COBOL.

```

,
INPUT "Enter statement: "; S$
FOR I = 1 TO LEN(S$)
  M$ = MID$(S$, I, 1): N$ = MID$(S$, I + 1, 1)
  MN$ = MID$(S$, I, 2)
  IF M$ = "=" OR M$ = ">" OR M$ = "<" THEN
    IF N$ = "=" OR N$ = ">" OR N$ = "<" THEN
      IF MN$ = "<=" OR MN$ = "=<" THEN PRINT "IS NOT GREATER
THAN";
      IF MN$ = ">=" OR MN$ = "=>" THEN PRINT "IS NOT LESS THAN";
      IF MN$ = "<>" OR MN$ = "><" THEN PRINT "IS NOT EQUAL TO";
      I = I + 1
    ELSE
      IF M$ = ">" THEN PRINT "IS GREATER THAN";
      IF M$ = "<" THEN PRINT "IS LESS THAN";
      IF M$ = "=" THEN PRINT "IS EQUAL TO";
    END IF
  ELSE
    PRINT M$;
  END IF
NEXT I

```

'2.4

' This program ranks teams in a league.

```

,
INPUT "Enter N: "; N
FOR I = 1 TO N
  INPUT "Enter team: "; N$(I)
  INPUT "Enter wins, losses: "; W(I), L(I)
NEXT I
FOR I = 1 TO N - 1
  FOR J = I + 1 TO N
    IF W(I) <= W(J) OR (W(I) = W(J) AND N$(I) > N$(J)) THEN
      SWAP W(I), W(J): SWAP L(I), L(J): SWAP N$(I), N$(J)
    END IF
  NEXT J
NEXT I
' Display teams in order
FOR I = 1 TO N
  IF W(I) = W(I - 1) THEN PRINT R; ELSE PRINT : PRINT I; : R = I
  PRINT N$(I); SPACE$(13 - LEN(N$(I))); W(I); ", "; L(I)
NEXT I

```

'2.5

' This program will guess a secret number within 7 tries.

```
'  
INC = 64: GUESS = 64  
WHILE A$ <> "R"  
  G = G + 1  
  PRINT USING "GUESS #"; G;  
  PRINT ":"; GUESS  
  INPUT "Enter H, L, or R: "; A$  
  INC = INC / 2  
  IF A$ = "L" THEN GUESS = GUESS - INC  
  IF A$ = "H" THEN GUESS = GUESS + INC  
WEND
```

'2.6

' This program prints text in pyramid form.

```
'  
INPUT "Enter text: "; A$: L = LEN(A$)  
I = 1  
WHILE I <= L  
  MD$ = MID$(A$, I, 1)  
  IF MD$ <> " " THEN  
    L$ = L$ + MD$  
  ELSE  
    IF LEN(L$) < PL + 2 THEN  
      L$ = L$ + MD$  
    ELSE  
      PL = LEN(L$)  
      PRINT SPACE$(20 - INT(PL / 2)); L$: L$ = ""  
    END IF  
  END IF  
  I = I + 1  
WEND  
PRINT SPACE$(20 - INT(LEN(L$) / 2)); L$
```

'2.7

' This program displays a rectangle of asterisks.

```
'  
INPUT "Enter length, width: "; L, W  
CLS  
COL = INT((80 - L) / 2): ROW = INT((24 - W) / 2)  
LOCATE ROW, COL  
FOR I = 1 TO L: PRINT "*"; : NEXT I  
FOR I = 1 TO W - 2  
  LOCATE ROW + I, COL: PRINT "*"   
  LOCATE ROW + I, COL + L - 1: PRINT "*"   
NEXT I  
LOCATE ROW + W - 1, COL  
FOR I = 1 TO L: PRINT "*"; : NEXT I
```

```
'2.8
' This program displays a bar graph for lengths.
,
DIM A(12)
INPUT "Enter title: "; T$
FOR I = 0 TO 11
  PRINT "Enter # for"; 1980 + I; ":";
  INPUT A(I): IF A(I) > MAX THEN MAX = A(I)
NEXT I
INC = MAX / 20
CLS : PRINT SPACE$(3); T$; SPACE$(3);
PRINT USING "ASTERISK = ####.##"; INC
FOR I = 20 TO 1 STEP -1: PRINT USING "##"; I: NEXT I
FOR I = 1 TO 12 * 3 + 2: PRINT "-"; : NEXT I
PRINT : PRINT " ";
FOR I = 0 TO 11: PRINT USING "###"; 80 + I; : NEXT I
FOR I = 0 TO 11
  FOR J = 1 TO INT(A(I) / INC)
    LOCATE 22 - J, I * 3 + 4: PRINT " *"
  NEXT J
NEXT I
LOCATE 23, 1
```

```
'2.9
' This program displays a store maintenance list.
,
INPUT "Enter # of entries in yesterday's file: "; F1
FOR I = 1 TO F1
  INPUT "Enter ID: "; ID1$(I)
  INPUT "Enter item: "; ITEM1$(I)
NEXT I
INPUT "Enter # of entries in today's file: "; F2
FOR I = 1 TO F2
  INPUT "Enter ID: "; ID2$(I)
  INPUT "Enter item: "; ITEM2$(I)
NEXT I
PRINT : PRINT "ADDED"
FOR I = 1 TO F2
  J = 1
  WHILE J < F1 AND ID2$(I) <> ID1$(J): J = J + 1: WEND
  IF ID2$(I) <> ID1$(J) THEN
    AN = AN + 1: PRINT ID2$(I); " "; ITEM2$(I)
  END IF
NEXT I
PRINT : PRINT "CHANGED"
FOR I = 1 TO F1
  J = 1
  WHILE J < F2 AND (ID1$(I) <> ID2$(J) OR ITEM1$(I) = ITEM2$(J))
    J = J + 1
  WEND
  IF ID1$(I) = ID2$(J) AND ITEM1$(I) <> ITEM2$(J) THEN
    CN = CN + 1: PRINT ID1$(I); " "; ITEM1$(I); " "; ITEM2$(J)
  END IF
NEXT I
PRINT : PRINT "DELETED"
FOR I = 1 TO F1
  J = 1
  WHILE J < F2 AND ID1$(I) <> ID2$(J): J = J + 1: WEND
  IF ID1$(I) <> ID2$(J) THEN
    DN = DN + 1: PRINT ID1$(I); " "; ITEM1$(I)
  END IF
NEXT I
PRINT
PRINT "TOTAL ADDED ="; AN
PRINT "TOTAL CHANGED ="; CN
PRINT "TOTAL DELETED ="; DN
```

```
'2.10
' This program displays the contents of contest diskettes.
,
INPUT "Enter year: "; Y$: YY$ = RIGHT$(Y$, 2): Y = VAL(YY$)
DATA PRB,JDG,PG1,PG2,BAS,PAS
FOR I = 1 TO 6: READ Z$(I): NEXT I
XXX$(1) = "ONE": XXX$(2) = "TWO": XXX$(3) = "THR"
FOR I = 1 TO 4
  FOR J = 1 TO 3
    PRINT "FHS"; YY$; "-"; MID$(STR$(J), 2); "."; Z$(I)
  NEXT J
NEXT I
TOT = 12
FOR I = 5 TO 6
  FOR J = 1 TO 3
    P = 10
    IF Y = 80 AND J = 3 THEN P = 12
    IF Y = 81 THEN P = 5
    IF Y = 82 AND J = 2 THEN P = 12
    IF Y = 82 AND J = 3 THEN P = 8
    FOR K = 1 TO P
      PRINT XXX$(J); MID$(STR$(K), 2); "T"; YY$; "."; Z$(I)
      TOT = TOT + 1
      IF TOT = 20 THEN
        A$ = "": WHILE A$ = "": A$ = INKEY$: WEND: TOT = 0
      END IF
    NEXT K
  NEXT J
NEXT I
```

```
'3.1
' This program simulates a baseball game.
,
DEFINT A-W
RANDOMIZE TIMER
CLS : PRINT
PRINT SPACE$(8);
FOR I = 1 TO 9: PRINT I; : NEXT I: PRINT " SCORE"
PRINT SPACE$(8); : FOR I = 1 TO 33: PRINT "-"; : NEXT I: PRINT
PRINT "TEAM A !"; SPACE$(27); "!"
PRINT "TEAM B !"; SPACE$(27); "!"
FOR IN = 1 TO 9
  FOR T = 1 TO 2
    S = 0: B = 0: W = 0: R = 0: O = 0
    WHILE O < 3
      X = RND(3)
      IF X < .4 THEN S = S + 1: STOT = STOT + 1
      IF X >= .4 THEN B = B + 1: BTOT = BTOT + 1
      IF S = 3 THEN O = O + 1: OTOT = OTOT + 1: S = 0: W = 0
      IF B = 4 THEN W = W + 1: WTOT = WTOT + 1: B = 0: S = 0
      IF W = 4 THEN R = R + 1: R(T) = R(T) + 1: W = 3
    WEND
    LOCATE 3 + T, 6 + IN * 3: PRINT R;
  NEXT T
NEXT IN
LOCATE 4, 39: PRINT USING "##"; R(1)
LOCATE 5, 39: PRINT USING "##"; R(2)
PRINT
PRINT "TOTAL # OF STRIKES:"; STOT
PRINT "TOTAL # OF BALLS:"; BTOT
PRINT "TOTAL # OF WALKS:"; WTOT
PRINT "TOTAL # OF STRIKE OUTS:"; OTOT
```

```
'3.2
' This program displays the units digit in a power expression.
,
DEFINT A-Z
INPUT "Enter A, X: "; A(1), X(1)
INPUT "Enter B, Y: "; A(2), X(2)
INPUT "Enter C, Z: "; A(3), X(3)
FOR I = 1 TO 3
  POW = 1
  FOR J = 1 TO X(I)
    POW = POW * A(I)
    C = INT(POW / 10)
    POW = POW - C * 10
  NEXT J
  SUM = SUM + POW
NEXT I
C = INT(SUM / 10)
PRINT SUM - C * 10
```

```
'3.3
' This program displays all digits in X ^ Y.
,
DEFINT A-Z
DIM A(200)
INPUT "Enter X, Y: "; X, Y
A(1) = 1: DIG = 1
FOR I = 1 TO Y
  FOR J = 1 TO DIG
    A(J) = A(J) * X + C
    C = INT(A(J) / 10)
    A(J) = A(J) - C * 10
  NEXT J
  WHILE C > 0
    CC = INT(C / 10): DIG = DIG + 1
    A(DIG) = C - CC * 10: C = CC
  WEND
NEXT I
FOR I = DIG TO 1 STEP -1
  PRINT USING "#"; A(I);
NEXT I
```

```

'3.4
' This program assigns user LOGON IDs to names.
'
INPUT "Enter name: "; N$(1): T = 1
WHILE N$(T) <> "END"
  T = T + 1
  INPUT "Enter name: "; N$(T)
WEND
' Extract parts of name for initials
T = T - 1
FOR I = 1 TO T
  FOR J = 1 TO LEN(N$(I))
    MD$ = MID$(N$(I), J, 1)
    IF MD$ <> " " THEN
      W$ = W$ + MD$
    ELSE
      IF F = 1 THEN M$(I) = W$: M = 1
      IF F = 0 THEN F$(I) = W$: F = 1
      W$ = ""
    END IF
  NEXT J
  IF M = 0 THEN M$(I) = "X"
  L$(I) = W$: W$ = "": F = 0: M = 0
  INIT$(I) = LEFT$(F$(I), 1) + LEFT$(M$(I), 1) + LEFT$(L$(I), 1)
  IN2$(I) = INIT$(I): N2$(I) = L$(I) + " " + F$(I): C(I) = I
NEXT I
' Sort Initials
FOR I = 1 TO T - 1
  FOR J = I + 1 TO T
    IF IN2$(I) > IN2$(J) THEN
      SWAP IN2$(I), IN2$(J): SWAP N2$(I), N2$(J): SWAP C(I), C(J)
    END IF
  NEXT J
NEXT I
' Sort names within same initials and assign numbers
J = 0
WHILE J < T - 1
  I = J + 1: J = I + 1
  WHILE (IN2$(I) <> IN2$(J)) AND (I < T)
    I = I + 1: J = J + 1
  WEND
  WHILE (IN2$(I) = IN2$(J)): J = J + 1: WEND: J = J - 1
  FOR A = I TO J - 1
    FOR B = A + 1 TO J
      IF N2$(A) > N2$(B) THEN
        SWAP N2$(A), N2$(B): SWAP C(A), C(B)
      END IF
    NEXT B
  NEXT A
' Assign numbers for middle initial
FOR A = I TO J
  MID$(INIT$(C(A)), 2, 1) = MID$(STR$(A - I + 1), 2, 1)
NEXT A
WEND
FOR I = 1 TO T

```

```

PRINT N$(I); SPACE$(19 - LEN(N$(I))); "SD"; INIT$(I); "1"
NEXT I

```

'3.5

```

' This program displays the digits 0 - 9 in enlarged form.
' 1 The data contains the
' 2 3 line segment #s (on the left)
' 4 that need to be displayed to
' 5 6 produce the corresponding
' 7 digits: 0,1,2,3,4,5,6,7,8,9
DATA 123567,36,13457,13467,2346,12467,124567,136,1234567,12346
FOR N = 0 TO 9
  CLS : READ A$
  FOR J = 1 TO LEN(A$)
    X = VAL(MID$(A$, J, 1))
    SELECT CASE X
      CASE 1: LOCATE 1, 1: PRINT STRING$(11, "*")
      CASE 2: FOR I = 1 TO 8: LOCATE I, 1: PRINT "*": NEXT I
      CASE 3: FOR I = 1 TO 8: LOCATE I, 11: PRINT "*": NEXT I
      CASE 4: LOCATE 8, 1: PRINT STRING$(11, "*")
      CASE 5: FOR I = 1 TO 8: LOCATE I + 7, 1: PRINT "*": NEXT I
      CASE 6: FOR I = 1 TO 8: LOCATE I + 7, 11: PRINT "*": NEXT I
      CASE 7: LOCATE 15, 1: PRINT STRING$(11, "*")
    END SELECT
  NEXT J
  SLEEP (1)
NEXT N

```

'3.6

```

' This program will evaluate an expression with ().
'
INPUT "Enter expression: "; A$
FOR I = 1 TO LEN(A$)
  M$ = MID$(A$, I, 1)
  IF M$ = "(" THEN P = P + 1: P1(P) = S + 1
  IF M$ = "+" OR M$ = "-" THEN S = S + 1: SY$(S) = M$
  IF M$ >= "0" AND M$ <= "9" THEN N = N + 1: NUM(N) = VAL(M$)
  IF M$ = ")" THEN
    FOR J = P1(P) TO S
      IF SY$(J) = "-" THEN NUM(J + 1) = NUM(J) - NUM(J + 1)
      IF SY$(J) = "+" THEN NUM(J + 1) = NUM(J) + NUM(J + 1)
    NEXT J
    N = P1(P): NUM(N) = NUM(S + 1)
    S = P1(P) - 1: P = P - 1
  END IF
NEXT I
FOR I = 1 TO S
  IF SY$(I) = "-" THEN NUM(I + 1) = NUM(I) - NUM(I + 1)
  IF SY$(I) = "+" THEN NUM(I + 1) = NUM(I) + NUM(I + 1)
NEXT I
PRINT NUM(N)

```

```

'3.7
' This program displays the two pay days for a given month.
,
DIM MON(12), MNAME$(12)
DATA MONDAY,TUESDAY,WEDNESDAY,THURSDAY,FRIDAY,SATURDAY,SUNDAY
DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY,AUGUST
DATA SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER
DATA 31,28,31,30,31,30,31,31,30,31,30,31
FOR I = 1 TO 7: READ DNAME$(I): NEXT I
FOR I = 1 TO 12: READ MNAME$(I): NEXT I
FOR I = 1 TO 12: READ MON(I): NEXT I
H = 1
INPUT "Enter holiday MM, DD: "; MHOL(H), DHOL(H)
WHILE MHOL(H) > 0
  H = H + 1
  INPUT "Enter holdiaay MM, DD: "; MHOL(H), DHOL(H)
WEND
H = H - 1
PRINT : INPUT "Enter month #: "; MNUM: PRINT
WHILE MNUM > 0
  DAYS(1) = 0
  FOR I = 1 TO MNUM - 1
    DAYS(1) = DAYS(1) + MON(I)
  NEXT I
  DAY(1) = 15: DAY(2) = MON(MNUM)
  DAYS(2) = DAYS(1) + DAY(2)
  DAYS(1) = DAYS(1) + DAY(1)
  FOR T = 1 TO 2
    HOL = 1
    ' Decrement days counters if holiday or weekend
    WHILE HOL = 1 OR WKEND = 1
      HOL = 0: WKEND = 0
      FOR I = 1 TO H
        IF MHOL(I) = MNUM AND DAY(T) = DHOL(I) THEN
          DAY(T) = DAY(T) - 1: DAYS(T) = DAYS(T) - 1: HOL = 1
        END IF
      NEXT I
      X = DAYS(T) MOD 7
      IF X = 5 OR X = 6 THEN          '5 = Saturday or 6 = Sunday
        DAY(T) = DAY(T) - 1: DAYS(T) = DAYS(T) - 1: WKEND = 1
      END IF
    WEND
    PRINT DNAME$(X + 1); " "; MNAME$(MNUM); DAY(T)
  NEXT T
  PRINT : INPUT "Enter month #: "; MNUM: PRINT
WEND

```

```

'3.8
' This program will display 3 x 3 magic squares.
'
INPUT "Enter digit: "; DIG
INPUT "Enter row, col: "; ROW, COL
DATA 6,7,2
DATA 1,5,9
DATA 8,3,4
FOR I = 1 TO 3: FOR J = 1 TO 3: READ A(I, J): NEXT J, I
ROT = 1
WHILE (A(ROW, COL) <> DIG) AND (ROT < 4)
' Rotate outer numbers clockwise, at most 3 times
  X = A(1, 1): A(1, 1) = A(3, 1): A(3, 1) = A(3, 3)
  A(3, 3) = A(1, 3): A(1, 3) = X
  X = A(1, 2): A(1, 2) = A(2, 1): A(2, 1) = A(3, 2)
  A(3, 2) = A(2, 3): A(2, 3) = X
  ROT = ROT + 1
WEND
IF A(ROW, COL) <> DIG THEN PRINT "NO SOLUTION": END
FOR P = 1 TO 2
  FOR I = 1 TO 3
    FOR J = 1 TO 3
      PRINT A(I, J);
    NEXT J: PRINT
  NEXT I: PRINT
  IF P = 1 THEN
    IF (ROW = 1 AND COL = 2) OR (ROW = 3 AND COL = 2) THEN
' Swap with respect to 2nd column
      SWAP A(1, 1), A(1, 3): SWAP A(2, 1), A(2, 3)
      SWAP A(3, 1), A(3, 3)
    END IF
    IF (ROW = 1 AND COL = 1) OR (ROW = 3 AND COL = 3) THEN
' Swap with respect to main diagonal
      SWAP A(1, 2), A(2, 1): SWAP A(1, 3), A(3, 1)
      SWAP A(3, 2), A(2, 3)
    END IF
    IF (ROW = 1 AND COL = 3) OR (ROW = 3 AND COL = 1) THEN
' Swap with respect to minor diagonal
      SWAP A(2, 1), A(3, 2): SWAP A(1, 1), A(3, 3)
      SWAP A(1, 2), A(2, 3)
    END IF
    IF (ROW = 2 AND COL = 1) OR (ROW = 2 AND COL = 3) THEN
' Swap with respect to 2nd row
      SWAP A(1, 1), A(3, 1): SWAP A(1, 2), A(3, 2)
      SWAP A(1, 3), A(3, 3)
    END IF
  END IF
NEXT P

```

```
'3.9
' This program will display a pie graph.
'
DIM A(21, 21)
INPUT "Enter 3 percentages: "; P(1), P(2), P(3)
A$(1) = "A": A$(2) = "D": A$(3) = "N"
CLS : PI = 3.14159
' Draw circle
FOR I = -PI / 2 TO 3 / 2 * PI STEP .1
  X = COS(I) * 10: Y = SIN(I) * 10
  LOCATE 11 + Y, 11 + X: PRINT "*": A(11 + Y, 11 + X) = 1
NEXT I
' Draw 3 line segments from center
FOR S = 0 TO 2
  SUM = SUM + P(S)
  I = -PI / 2 + 2 * PI * SUM / 100
  FOR R = 0 TO 10
    X = COS(I) * R: Y = SIN(I) * R
    LOCATE 11 + Y, 11 + X: PRINT "*": A(11 + Y, 11 + X) = 1
  NEXT R
NEXT S
A$ = INPUT$(1): SUM = 0
' Fill regions with letters
FOR S = 1 TO 3
  LSUM = SUM: SUM = SUM + P(S)
  FOR L = LSUM TO SUM
    I = -PI / 2 + 2 * PI * L / 100
    FOR R = 1 TO 9
      X = COS(I) * R: Y = SIN(I) * R
      IF A(11 + Y, 11 + X) = 0 THEN
        LOCATE 11 + Y, 11 + X: PRINT A$(S)
      END IF
    NEXT R
  NEXT L
NEXT S
```

```

'3.10
' This program will convert large numbers in base 2,4,8,16.
,
DEFINT A-Z
DIM A(255)
INPUT "Enter numeral: "; NUM$
INPUT "Enter base M: "; M
INPUT "Enter base N: "; N
L = LEN(NUM$)
DIGM = INT(LOG(M) / LOG(2) + .001)
DIGN = INT(LOG(N) / LOG(2) + .001)
PAD = DIGN - (DIGM * L MOD DIGN): IF PAD = DIGN THEN PAD = 0
FOR I = 1 TO PAD: A(I) = 0: NEXT I
' Convert from base M to base 2
FOR I = 1 TO L
  D$ = MID$(NUM$, I, 1)
  NUM = INSTR("0123456789ABCDEF", D$) - 1
  FOR J = DIGM - 1 TO 0 STEP -1
    X = INT(NUM / 2 ^ J)
    IND = I * DIGM - J + PAD
    A(IND) = X
    NUM = NUM - X * 2 ^ J
  NEXT J
NEXT I
' Convert from base 2 to base N
LIND = DIGM * L + PAD: ZERO = 1
FOR I = 0 TO (LIND / DIGN) - 1
  SUM = 0
  FOR J = 1 TO DIGN
    IND = I * DIGN + J
    SUM = SUM + A(IND) * 2 ^ (DIGN - J)
  NEXT J
  IF ZERO = 0 OR SUM > 0 THEN
    ZERO = 0
    PRINT MID$("0123456789ABCDEF", SUM + 1, 1);
  END IF
NEXT I

```

FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '92
BASIC PROGRAM SOLUTIONS

'1.1

' This program displays the company name: GTEDS.

'

```
PRINT "GGGGG   TTTTT   EEEEE"
PRINT "G       T       E"
PRINT "G GGG   T       EEEEE   DATA SERVICES"
PRINT "G  G    T       E"
PRINT "GGGGG   T       EEEEE"
```

'1.2

' This program will display the company name in a year.

'

```
INPUT "Enter year:"; YEAR
IF YEAR < 1920 THEN
  PRINT "RICHLAND CENTER TELEPHONE COMPANY"
ELSE
  IF YEAR < 1926 THEN
    PRINT "COMMONWEALTH TELEPHONE COMPANY"
  ELSE
    IF YEAR < 1935 THEN
      PRINT "ASSOCIATED TELEPHONE UTILITIES COMPANY"
    ELSE
      IF YEAR < 1959 THEN
        PRINT "GENERAL TELPHONE CORPORATION"
      ELSE
        IF YEAR < 1982 THEN
          PRINT "GENERAL TELPHONE & ELECTRONICS CORPORATION"
        ELSE
          PRINT "GTE CORPORATION"
        END IF
      END IF
    END IF
  END IF
END IF
```

'1.3

' This program will determine company's ranking in Forbes.

'

```
INPUT "Enter 1991 rank:"; RANK
INPUT "Enter number of places:"; PLACES
PRINT RANK - PLACES
```

'1.4

' This program will indent GTE's 6 operations.

```
,
INPUT "Enter number of spaces:"; X
PRINT "GTE TELEPHONE OPERATIONS"
PRINT SPACE$(X); "GTE GOVERNMENT SYSTEMS"
PRINT SPACE$(X * 2); "GTE MOBILE COMMUNICATIONS"
PRINT SPACE$(X * 3); "GTE INFORMATION SERVICES"
PRINT SPACE$(X * 4); "GTE SPACENET"
PRINT SPACE$(X * 5); "GTE AIRPHONE"
```

'1.5

' This program will display # of WHOLE YEARS GTEDS existed.

```
,
INPUT "Enter M, Y:"; M, Y
IF M < 10 THEN X = 1 ELSE X = 0
PRINT Y - 1967 - X; "YEARS"
```

'1.6

' This program will center a title and name in a box.

```
,
INPUT "Enter title:"; T$
INPUT "Enter name:"; N$
PRINT STRING$(24, "*")
PRINT "*"; SPACE$(22); "*"
L = LEN(T$) + LEN(N$) + 1
SP1 = INT((22 - L) / 2)
SP2 = (22 - L) - SP1
PRINT "*"; SPACE$(SP1); T$; " "; N$; SPACE$(SP2); "*"
PRINT "*"; SPACE$(22); "*"
PRINT STRING$(24, "*")
```

'1.7

' This program will display a 4-line statement for ISOP.

```
,
INPUT "Enter name:"; N$
INPUT "Enter title:"; T$
INPUT "Enter group:"; G$
PRINT N$; " IS A "; T$; " WITHIN THE"
PRINT G$; " GROUP AND"
PRINT "HAS BEEN SELECTED TO PARTICIPATE IN"
PRINT "THE ISOP."
```

'1.8

' This program will display a dollar sign next to an amount.

```
,
INPUT "Enter amount: "; AMOUNT$
A = VAL(AMOUNT$)
IF A >= 2000 THEN PRINT "$2000.00" ELSE PRINT "$"; AMOUNT$
```

'1.9

' This program will display an acronym for business words.

,

```
INPUT "Enter words: "; ST$
```

```
PRINT LEFT$(ST$, 1);
```

```
FOR I = 2 TO LEN(ST$) - 1
```

```
  IF MID$(ST$, I, 1) = " " THEN PRINT MID$(ST$, I + 1, 1);
```

```
NEXT I
```

'1.10

' This program will calculate QUALITY hours and minutes.

,

```
INPUT "Enter number of technicians, N:"; N
```

```
INPUT "Enter number of minutes, M:"; M
```

```
TOTAL = 50 * 5 * N * M
```

```
HOURS = INT(TOTAL / 60)
```

```
MIN = TOTAL - HOURS * 60
```

```
PRINT HOURS; "HOURS"; MIN; "MINUTES"
```

```
'2.1
' This program will display a speech indented.
,
I = 0
WHILE (LINE$(I) > "") OR (I = 0)
  I = I + 1
  INPUT "Enter line:"; LINE$(I)
WEND
FOR J = 1 TO I - 1
  CH$ = MID$(LINE$(J), 1, 1)
  IF CH$ = "I" THEN PRINT LINE$(J)
  IF CH$ >= "A" AND CH$ <= "H" THEN PRINT SPACE$(4); LINE$(J)
  IF VAL(CH$) > 0 THEN PRINT SPACE$(8); LINE$(J)
NEXT J
```

```
'2.2
' This program will display a number in words.
,
DIM WORDS$(27)
DATA ONE, TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN
DATA ELEVEN, TWELVE, THIRTEEN, FOURTEEN, FIFTEEN, SIXTEEN
DATA SEVENTEEN, EIGHTEEN, NINETEEN, TWENTY, THIRTY, FORTY
DATA FIFTY, SIXTY, SEVENTY, EIGHTY, NINETY
FOR I = 1 TO 27: READ WORDS$(I): NEXT I
INPUT "Enter number:"; NUM
IF NUM < 20 THEN PRINT WORDS$(NUM): END
TENS = INT(NUM / 10)
UNITS = NUM - TENS * 10
PRINT WORDS$(18 + TENS);
IF UNITS > 0 THEN PRINT "-"; WORDS$(UNITS)
```

'2.3

' This program will display selected items from a NRD menu.

```

,
DATA "DEMONSTRATED INTEREST IN INFORMATION MANAGMENT."
DATA "DEMONSTRATED LEADERSHIP SKILLS."
DATA "STRONG GPA/PERFORMANCE HISTORY."
DATA "AT LEAST TWO COURSES IN ANY PROGRAMMING LANGUAGE."
DATA "INTERNSHIP OR WORK EXPERIENCE."
DATA "EFFECTIVE ORAL AND WRITTEN COMMUNICATION SKILLS."
DATA "CAREER DEVELOPMENT POTENTIAL."
FOR I = 1 TO 7: READ CRIT$(I): NEXT I
INPUT "Enter name:"; NAM$
INPUT "Enter degree:"; DEGREE$
FOR I = 1 TO 7
  PRINT USING "#. "; I; : PRINT CRIT$(I)
NEXT I
PRINT
INPUT "Select up to 7 items:"; ITEMS$
CLS
PRINT NAM$: PRINT DEGREE$
NUM = 0
FOR I = 1 TO 7
  I$ = LTRIM$(STR$(I))
  IF INSTR(1, ITEMS$, I$) > 0 THEN
    NUM = NUM + 1
    PRINT : PRINT USING "#. "; NUM; : PRINT CRIT$(I)
  END IF
NEXT I

```

'2.4

' This program will rate a speech.

```

,
DATA SPEECH VALUE, PREPARATION, MANNER, ORGANIZATION
DATA OPENING, BODY OF SPEECH, CONCLUSION
FOR I = 1 TO 7: READ CAT$(I): NEXT I
DATA EXCELLENT, ABOVE AVERAGE, SATISFACTORY
DATA SHOULD IMPROVE, MUST IMPROVE
FOR I = 1 TO 5: READ VERBAL$(I): NEXT I
FOR I = 1 TO 7
  PRINT "Enter rating for "; CAT$(I);
  INPUT ": "; RATING$(I)
NEXT I
FOR I = 1 TO 7
  NUM = 1
  WHILE (RATING$(I) <> VERBAL$(NUM)) AND (NUM < 7)
    NUM = NUM + 1
  WEND
  PRINT CAT$(I); ":"; NUM
  TOTAL = TOTAL + NUM
NEXT I
200 PRINT
210 AVE = TOTAL / 7
220 PRINT USING "AVERAGE NUMERICAL RATING = #.#"; AVE
230 PRINT "SPEECH RATING = "; VERBAL$(INT(AVE + .5))

```

'2.5

' This program will format GTEDS MISSION statement.

```

,
DATA "BE THE CUSTOMER-ORIENTED LEADER AND PROVIDER-OF-CHOICE "
DATA "OF QUALITY INFORMATION PRODUCTS AND SERVICES IN THE "
DATA "TELECOMMUNICATIONS MARKETPLACE AND SELECTED OTHER "
DATA "RELATED MARKETS IN SUPPORT OF GTE'S TELOPS GOALS."
FOR I = 1 TO 4: READ ST$(I): NEXT I
INPUT "Enter N:"; N
STATES$ = ST$(1) + ST$(2) + ST$(3) + ST$(4)
FOR I = 1 TO LEN(STATES$)
  CH$ = MID$(STATES$, I, 1)
  WORD$ = WORD$ + CH$
  IF (CH$ = " " OR CH$ = "-" OR CH$ = ".") THEN
    NUMCH = LEN(LINE$) + LEN(WORD$)
    IF CH$ = " " THEN NUMCH = NUMCH - 1
    IF NUMCH > N THEN PRINT LINE$: LINE$ = WORD$
    IF NUMCH <= N THEN LINE$ = LINE$ + WORD$
    WORD$ = ""
  END IF
NEXT I
PRINT LINE$; WORD$

```

'2.6

' This program will change (.) to (?) at end of sentence.

```

,
DATA WHAT,WHY,HOW,WHO,WHERE
FOR I = 1 TO 5: READ QUEST$(I): NEXT I
INPUT "Enter paragraph:"; PAR$: PRINT
FIRSTW = -1
FOR I = 1 TO LEN(PAR$)
  CH$ = MID$(PAR$, I, 1)
  IF CH$ = " " AND LEN(FIRSTW$) > 0 THEN
    FIRSTW = 0
  ELSE
    IF (CH$ = "." OR CH$ = "!" OR CH$ = "?") THEN
      IF CH$ = "." THEN
        FOR J = 1 TO 5
          IF FIRSTW$ = QUEST$(J) THEN CH$ = "?"
        NEXT J
      END IF
      FIRSTW$ = "": FIRSTW = -1
    ELSE
      IF FIRSTW AND (CH$ <> " ") THEN FIRSTW$ = FIRSTW$ + CH$
    END IF
  END IF
PRINT CH$;
NEXT I

```

'2.7

' This program will print names in the office at a beep.

,

```
DATA DAVID,0700,1600
DATA DON,0800,1700
DATA DOUG,0730,1630
DATA GRANDVILLE,1230,2100
DATA JAMES,1130,2200
DATA JIM,0900,1800
DATA JOHN,0700,1600
DATA LINDA,1230,2300
DATA MARIE,0700,1600
DATA MATT,1230,2300
DATA PAULA,0700,1600
DATA ROBERT,0800,1700
DATA SHELLEY,0630,1530
DATA TOM,1100,1930
DIM NAM$(14), START(14), QUIT(14)
FOR I = 1 TO 14
  READ NAM$(I), START(I), QUIT(I)
NEXT I
INPUT "Enter time:"; TIME
INPUT "Enter day:"; DAY$
FOR I = 1 TO 14
  IF (START(I) <= TIME) AND (TIME <= QUIT(I)) THEN
    IF (DAY$ <> "SUNDAY") AND (DAY$ <> "SATURDAY") THEN
      INOFFICE = -1
      IF (NAM$(I) = "JAMES") AND (DAY$ = "MONDAY") THEN INOFFICE = 0
      IF (NAM$(I) = "LINDA") AND (DAY$ = "FRIDAY") THEN INOFFICE = 0
      IF (NAM$(I) = "MATT") AND (DAY$ = "MONDAY") THEN INOFFICE = 0
      IF INOFFICE THEN
        NUM = NUM + 1
        IF NUM = 1 THEN PRINT NAM$(I);
        IF NUM > 1 THEN PRINT ", "; NAM$(I);
      END IF
    END IF
  END IF
NEXT I
IF NUM = 0 THEN PRINT "NONE"
```

```
'2.8
' This program will randomly assign titles to a team.
,
DATA WILL,DARLENE,JEFF,LIZ,LORI,MARY,PING
FOR I = 1 TO 7: READ NAM$(I): NEXT I
DATA AUTHOR,MODERATOR,READER,RECORDER,INSPECTOR
FOR I = 1 TO 5: READ TITLE$(I): NEXT I
RANDOMIZE TIMER
INPUT "Enter author's name:"; TNAME$(1)
' Choose moderator
IF TNAME$(1) = NAM$(1) THEN
  TNAME$(2) = NAM$(2)
ELSE
  IF TNAME$(1) = NAM$(2) THEN
    TNAME$(2) = NAM$(1)
  ELSE
    TNAME$(2) = NAM$(INT(RND(3) * 2) + 1)
  END IF
END IF
' Choose next 3 title names
FOR I = 3 TO 5
  VALID = 0
  WHILE NOT VALID
    VALID = -1
    X = INT(RND(3) * 7) + 1
    FOR J = 1 TO I
      IF NAM$(X) = TNAME$(J) THEN VALID = 0
    NEXT J
  WEND
  TNAME$(I) = NAM$(X)
NEXT I
' Display all 5 titles and names.
FOR I = 1 TO 5
  PRINT TITLE$(I); " - "; TNAME$(I)
NEXT I
```

'2.9

' This program will sort a list of names with area codes.

,

```
DIM NAM$(15)
INPUT "Enter two area codes:"; AREA1, AREA2
INPUT "Enter number of names:"; NUM
FOR I = 1 TO NUM
  INPUT "Enter name:"; NAM$(I)
NEXT I
FOR I = 1 TO NUM - 1
  FOR J = I + 1 TO NUM
    IF NAM$(I) > NAM$(J) THEN SWAP NAM$(I), NAM$(J)
  NEXT J
NEXT I
IF AREA1 > AREA2 THEN A = AREA1: AREA1 = AREA2: AREA2 = A
MID = INT((NUM + 1) / 2)
FOR I = 1 TO MID
  PRINT AREA1; "- "; NAM$(I)
NEXT I
FOR I = MID + 1 TO NUM
  PRINT AREA2; "- "; NAM$(I)
NEXT I
```

```

'2.10
' This program will adjust a golf score by handicap.
,
DATA 5,4,4,4,3,4,4,3,5
FOR I = 1 TO 9: READ PAR(I): NEXT I
INPUT "Enter handicap:"; HAND
PRINT "Enter gross scores:";
INPUT G(1), G(2), G(3), G(4), G(5), G(6), G(7), G(8), G(9)
PRINT "HOLE #:";
FOR I = 1 TO 9: PRINT USING "####"; I; : NEXT I
PRINT : PRINT "PAR:  ";
FOR I = 1 TO 9
  PRINT USING "####"; PAR(I);
  PARTOT = PARTOT + PAR(I)
NEXT I
PRINT : PRINT "GROSS: ";
FOR I = 1 TO 9
  PRINT USING "####"; G(I);
  GTOT = GTOT + G(I)
NEXT I
PRINT : PRINT "ADJUST:";
' Determine # of tripple and double bogeys allowed
IF HAND > 9 THEN BOG(3) = HAND - 9: BOG(2) = 9 - BOG(3)
IF HAND <= 9 THEN BOG(2) = HAND: BOG(1) = 9 - BOG(2)
' Adjust the gross scores by Handicap
FOR I = 1 TO 9
  DIFF = G(I) - PAR(I)
  ADJUSTED = 0
  B = 3
  WHILE NOT ADJUSTED AND (B > 0)
    IF (BOG(B) > 0) AND (DIFF >= B) THEN
      A(I) = PAR(I) + B
      BOG(B) = BOG(B) - 1
      ADJUSTED = -1
    END IF
    B = B - 1
  WEND
  IF NOT ADJUSTED THEN A(I) = G(I)
NEXT I
' Display the adjusted scores and totals
FOR I = 1 TO 9
  PRINT USING "####"; A(I);
  ATOT = ATOT + A(I)
NEXT I
PRINT : PRINT
PRINT "PAR TOTAL:"; PARTOT
PRINT "GROSS TOTAL:"; GTOT
PRINT "ADJUST TOTAL:"; ATOT
PRINT "ROUND HANDICAP:"; ATOT - PARTOT

```

```
'3.1
' This program will move a triangle of GTEDS around the screen.
,
DATA "          "
DATA "      G    "
DATA "    T T    "
DATA "  E      E  "
DATA " D        D "
DATA " SDETGTEDS "
DATA "          "
FOR I = 1 TO 7: READ A$(I): NEXT I
CLS
ROW = 9: COL = 34
WHILE CH$ <> CHR$(27)
  FOR I = 1 TO 7
    LOCATE ROW + I, COL: PRINT A$(I);
  NEXT I
  C$ = INKEY$: IF C$ > "" THEN CH$ = C$
  FOR I = 1 TO 100: NEXT I
  SELECT CASE UCASE$(CH$)
    CASE "I": ROW = ROW - 1
    CASE "M": ROW = ROW + 1
    CASE "J": COL = COL - 1
    CASE "K": COL = COL + 1
  END SELECT
  IF ROW = 0 THEN ROW = 1: CH$ = ""
  IF COL = 0 THEN COL = 1: CH$ = ""
  IF ROW = 18 THEN ROW = 17: CH$ = ""
  IF COL = 69 THEN COL = 68: CH$ = ""
WEND
```

'3.2

' This program will display a date in 1992 after # of days.

,

```
DIM MONTH(12), MNAME$(12)
DATA TUESDAY,WEDNESDAY,THURSDAY,FRIDAY,SATURDAY,MONDAY
FOR I = 1 TO 6: READ DAY$(I): NEXT I
DATA 31,29,31,30,31,30,31,31,30,31,30,31
FOR I = 1 TO 12: READ MONTH(I): NEXT I
DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY,AUGUST
DATA SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER
FOR I = 1 TO 12: READ MNAME$(I): NEXT I
INPUT "Enter X:"; X
X = X + 1
D = (X MOD 6) + 1
PRINT DAY$(D); " ";
X = X + INT((X + 1) / 6)
I = 1
WHILE SUM + MONTH(I) < X
    SUM = SUM + MONTH(I): I = I + 1
WEND
PRINT MNAME$(I); X - SUM
IF DAY$(D) <> "SATURDAY" THEN END
X = X + 1
WHILE SUM + MONTH(I) < X
    SUM = SUM + MONTH(I): I = I + 1
WEND
PRINT "SUNDAY "; MNAME$(I); X - SUM
```

```
'3.3
' This program will release program modules for PWS.
,
WHILE NOT ALLDONE
  I = NUM + 1
  INPUT "Enter name, program: "; NAME$(I), PROG$(I)
  ' Find previous Name/Prog or make addition
  J = 1
  NOTFOUND = (NAME$(J) <> NAME$(I) OR PROG$(J) <> PROG$(I))
  WHILE (J < I) AND NOTFOUND
    J = J + 1
    NOTFOUND = (NAME$(J) <> NAME$(I) OR PROG$(J) <> PROG$(I))
  WEND
  I = J
  IF I > NUM THEN NUM = I
  INPUT "Enter completed, release: "; COMP$(I), REL$(I)
  IF REL$(I) = "Y" THEN COMP$(I) = "Y"
  MODCOMP = (COMP$(I) = "Y")
  ' Check if Module completed by all, and at least 1 released
  IF MODCOMP THEN
    MODREL = 0
    FOR J = 1 TO NUM
      IF PROG$(J) = PROG$(I) THEN
        IF COMP$(J) <> "Y" THEN MODCOMP = 0
        IF REL$(J) = "Y" THEN MODREL = -1
      END IF
    NEXT J
  ' If Module completed by all and 1 or more released
  IF (MODCOMP AND MODREL) THEN
    PRINT "MODULE "; PROG$(I); " HAS BEEN RELEASED"
    MODULE$ = PROG$(I)
    FOR J = 1 TO NUM
      IF PROG$(J) = MODULE$ THEN PROG$(J) = ""
    NEXT J
    ALLDONE = -1
    FOR J = 1 TO NUM
      IF PROG$(J) <> "" THEN ALLDONE = 0
    NEXT J
  END IF
END IF
WEND
```

'3.4

' This program will produce acronyms for phone numbers.

,

DIM A\$(18), B\$(18)

DATA AGENT, SOAP, MONEY, JEWEL, BALL, LOANS, CARE, SAVE, CALL, PAVE

DATA KEEP, KINGS, KNIFE, KNOCK, JOINT, JUICE, LOBBY, RATE

FOR I = 1 TO 18: READ B\$(I): A\$(I) = B\$(I): NEXT I

L1\$ = " ADGJMPTW"

L2\$ = " BEHKNRUX"

L3\$ = " CFILOSVY"

' Sort the data alphabetically

FOR I = 1 TO 17

 FOR J = I + 1 TO 18

 IF A\$(I) > A\$(J) THEN SWAP A\$(I), A\$(J)

 NEXT J

NEXT I

,

INPUT "Enter phone #:"; PH\$

P4\$ = MID\$(PH\$, 5, 4): P5\$ = MID\$(PH\$, 3, 1) + P4\$

' Convert words to number strings

FOR I = 1 TO 18

 L = LEN(A\$(I)): NUM\$ = ""

 FOR J = 1 TO L

 K = 1: C\$ = MID\$(A\$(I), J, 1): NOMATCH = -1

 WHILE NOMATCH

 K = K + 1

 IF MID\$(L1\$, K, 1) = C\$ THEN NOMATCH = 0

 IF MID\$(L2\$, K, 1) = C\$ THEN NOMATCH = 0

 IF MID\$(L3\$, K, 1) = C\$ THEN NOMATCH = 0

 WEND

 NUM\$ = NUM\$ + CHR\$(48 + K)

 NEXT J

 IF L = 4 AND NUM\$ = P4\$ THEN PRINT MID\$(PH\$, 1, 4); A\$(I)

 IF L = 5 AND NUM\$ = P5\$ THEN

 PRINT MID\$(PH\$, 1, 2); MID\$(A\$(I), 1, 1); "-";

 PRINT MID\$(A\$(I), L - 3, 4)

 END IF

NEXT I

```

'3.5
' This program will find seven 7-digit squares in base 8.
'
NUM = 1242: SNUM = 0
WHILE SNUM < 7
  NUM1$ = MID$(STR$(NUM), 2)
  ' Convert NUM1$ to base 10 number NUM1V
  NUM1V = 0
  FOR I = 1 TO 4
    DIGIT = ASC(MID$(NUM1$, I, 1)) - ASC("0")
    POWER = 1
    FOR J = 1 TO LEN(NUM1$) - I
      POWER = POWER * 8
    NEXT J
    NUM1V = NUM1V + DIGIT * POWER
  NEXT I
  NUM1V = NUM1V * NUM1V
  SQUARE$ = "": VALID = -1
  FOR I = 0 TO 7: DUP(I) = 0: NEXT I
  ' Convert Num1V to Base8 number
  J = INT(LOG(NUM1V) / LOG(8))
  WHILE (J >= 0) AND VALID
    POWER = 1
    FOR K = 1 TO J: POWER = POWER * 8: NEXT K
    X = INT(NUM1V / POWER)
    ' Check for duplicate digits
    IF DUP(X) THEN
      VALID = 0
    ELSE
      DUP(X) = -1
      SQUARE$ = SQUARE$ + CHR$(48 + X)
      NUM1V = NUM1V - X * POWER
    END IF
    J = J - 1
  WEND
  IF VALID THEN SNUM = SNUM + 1: PRINT SQUARE$; " "; NUM
  ' Increment to next base 8 number
  NUM = NUM + 1: NUMST$ = LTRIM$(STR$(NUM))
  WHILE INSTR(1, NUMST$, "8") > 0 OR INSTR(1, NUMST$, "9") > 0
    NUM = NUM + 1
    NUMST$ = LTRIM$(STR$(NUM))
  WEND
WEND

```

'3.6

' This program will find 3 distinct integers that are pairwise
' relatively prime such that they sum to N.

```
INPUT "Enter N:"; N
X = 2 + (N MOD 2)
WHILE (X < INT(N / 3)) AND NOT FOUND
  Y = X + 1
  WHILE (Y < INT((N - X) / 2)) AND NOT FOUND
    Z = N - X - Y: FOUND = -1
    FOR I = 2 TO Y
      IF (X MOD I = 0) AND (Y MOD I = 0) THEN FOUND = 0
      IF (X MOD I = 0) AND (Z MOD I = 0) THEN FOUND = 0
      IF (Y MOD I = 0) AND (Z MOD I = 0) THEN FOUND = 0
    NEXT I
    IF FOUND THEN PRINT X; "+"; Y; "+"; Z; "="; N
    IF NOT FOUND THEN Y = Y + 1
  WEND
  Z = Z + 1
WEND
```

```
'3.7
' This program will print combinations of 6 soccer players.
,
DATA ANDY,DAN,DOUG,JACK,MIKE,YEHIA
FOR I = 1 TO 6: READ NAME$(I): NEXT I
INPUT "Enter number of substitutes:"; NUMOFSUB
L = 6 + NUMOFSUB
FOR I = 7 TO L
  INPUT "Enter name:"; NAME$(I)
NEXT I
' Sort names with substitutes
FOR I = 1 TO L - 1
  FOR J = I + 1 TO L
    IF NAME$(I) >= NAME$(J) THEN SWAP NAME$(I), NAME$(J)
  NEXT J
NEXT I
,
M = 6
FOR I = 1 TO M: A(I) = M - I + 1: NEXT I
N = 1: A(1) = A(1) - 1
WHILE N <= M
  A(N) = A(N) + 1
  IF N > 1 THEN
    FOR I = N - 1 TO 1 STEP -1: A(I) = A(I + 1) + 1: NEXT I
  END IF
  IF A(N) <= L - N + 1 THEN
    S = S + 1
    PRINT S; NAME$(A(M));
    FOR I = M - 1 TO 1 STEP -1
      PRINT ", "; NAME$(A(I));
    NEXT I
    PRINT
    N = 0
    IF S MOD 24 = 0 THEN WHILE INKEY$ = "": WEND
  END IF
  N = N + 1
WEND
```

```

'3.8
' This program displays the Bill Date and the Due Date.
' January 1, 1992 was a Wednesday
'
DIM MNAME$(12), MON(12), MHOL(12), DHOL(12)
DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY,AUGUST
DATA SEPTEMBER,OCTOBER,NOVEMBER,DECEMBER
FOR I = 1 TO 12: READ MNAME$(I): NEXT I
DATA 31,29,31,30,31,30,31,31,30,31,30,31
FOR I = 1 TO 12: READ MON(I): NEXT I
DATA TUESDAY,WEDNESDAY,THURSDAY,FRIDAY,SATURDAY,SUNDAY,MONDAY
FOR I = 1 TO 7: READ DNAME$(I): NEXT I
INPUT "Enter month of bill:"; MNUM
INPUT "Enter cycle number:"; CYCLE
INPUT "Enter number of days:"; NUMDAYS
H = 1
INPUT "Enter holiday MM, DD:"; MHOL(H), DHOL(H)
WHILE MHOL(H) > 0
    H = H + 1
    INPUT "Enter holiday MM, DD:"; MHOL(H), DHOL(H)
WEND
H = H - 1: PRINT
DAYS(1) = 0
FOR I = 1 TO MNUM - 1
    DAYS(1) = DAYS(1) + MON(I)
NEXT I
DAY(1) = 3 * CYCLE - 2: DAY(2) = DAY(1) + NUMDAYS
DAYS(2) = DAYS(1) + DAY(2)
DAYS(1) = DAYS(1) + DAY(1)
FOR T = 1 TO 2
    HOL = 1: WKEND = 1
    ' Decrement days counter if holiday or weekend
    WHILE (HOL = 1) OR (WKEND = 1)
        HOL = 0: WKEND = 0
        IF DAY(T) > MON(MNUM) THEN
            DAY(T) = DAY(T) - MON(MNUM)
            MNUM = MNUM + 1
        END IF
        FOR I = 1 TO H
            IF MHOL(I) = MNUM AND DHOL(I) = DAY(T) THEN
                DAY(T) = DAY(T) + 1
                DAYS(T) = DAYS(T) + 1: HOL = 1
            END IF
        NEXT I
        X = DAYS(T) MOD 7
        IF (X = 4) OR (X = 5) THEN
            ' Saturday or Sunday
            DAY(T) = DAY(T) + 1
            DAYS(T) = DAYS(T) + 1: WKEND = 1
        END IF
    WEND
    IF T = 1 THEN PRINT "BILL "; ELSE PRINT "DUE ";
    PRINT "DATE: "; DNAME$(X + 1); " "; MNAME$(MNUM); DAY(T)
NEXT T

```

'3.9

' This program will calculate the area of a polygon room.

,

INPUT "Enter number of sides:"; SIDES

FOR I = 1 TO SIDES

 INPUT "Enter movement:"; MOV\$

 DIR\$(I) = MID\$(MOV\$, 1, 1)

 L = LEN(MOV\$)

 MOV\$ = MID\$(MOV\$, 2, L - 1)

 DIST(I) = VAL(MOV\$)

' Subtract Down and Left directions

 IF DIR\$(I) = "D" OR DIR\$(I) = "L" THEN DIST(I) = -DIST(I)

NEXT I

' Multiply length by width to obtain rectangle area,

' then add or subtract area from overall area.

I = 1: SUM = 0: AREA = 0

WHILE (I <= SIDES)

 SUM = SUM + DIST(I)

 AREA = AREA + (SUM * DIST(I + 1))

 I = I + 2

WEND

PRINT "AREA ="; ABS(AREA); "SQARE FEET"

```

'3.10
' This program will display the reasons a Rubik's Cube is
' unsolvable. Input is to be separated by a space (not a ,)
'
DATA "TOP:  ", "FRONT: ", "RIGHT: ", "BACK:  ", "LEFT:  "
DATA "BOTTOM:"
FOR I = 1 TO 6: READ SIDE$(I): NEXT I
EDGES$ = "T2P2 T6R2 T8F2 T4L2 F4L6 F6R4 "
EDGES$ = EDGES$ + "R6P4 P6L4 F8B2 R8B6 P8B8 L8B4"
FOR I = 1 TO 6
  PRINT "Enter colors on "; SIDE$(I);
  INPUT COLORS$
  FOR J = 1 TO 9
    COL$(I, J) = MID$(COLORS$, J * 2 - 1, 1)
  NEXT J
NEXT I
'
MIDUNIQUE = -1
FOR I = 1 TO 5
  FOR J = I + 1 TO 6
    IF COL$(I, 5) = COL$(J, 5) THEN MIDUNIQUE = 0
  NEXT J
NEXT I
'
IF NOT MIDUNIQUE THEN
  PRINT "COLORS ON MIDDLE SQUARES ARE NOT UNIQUE"
END IF
FOR K = 1 TO 12
  S1 = INSTR(1, "TFRPLB", MID$(EDGES$, K * 5 - 4, 1))
  N1 = ASC(MID$(EDGES$, K * 5 - 3, 1)) - ASC("0")
  S2 = INSTR(1, "TFRPLB", MID$(EDGES$, K * 5 - 2, 1))
  N2 = ASC(MID$(EDGES$, K * 5 - 1, 1)) - ASC("0")
  IF COL$(S1, N1) = COL$(S2, N2) THEN ENUM = ENUM + 1
NEXT K
PRINT "NUMBER OF EDGE PIECES HAVING SAME COLOR: "; ENUM

```

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:"; TITLES$
  NAM$(I) = NAM$(I) + " - " + TITLES$
  J = 1
  WHILE TITLES$(J) <> TITLES$: 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 format 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
  END IF
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:"; DIGIT$
INPUT "Enter sum:"; SUM
NEWSUM = INT(SUM / 10) * 8 + (SUM MOD 10)
LAST = LEN(DIGIT$)
FOR I = 1 TO LAST
  DIGIT(I) = VAL(MID$(DIGIT$, 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 divisible by Prime
  NUM = 0
  FOR I = 1 TO L
    NUM = NUM * 10 + A(I)
    Q(I) = INT(NUM / PRIME)
  NEXT I
  IF NUM = 0 THEN
    PRIME = PRIME + 1
    FIRSTFACTOR = FIRSTFACTOR + 1
    QUOTIENTIS0 = 1
  END IF
END WHILE

```

```

    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
    QUOTIENTISO = (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

```

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$)
PLAN A = 9999: PLAN B = 9999: PLAN C = 9999
IF (CALLLEN >= 5!) AND DIFFAREA THEN
    PLAN A = COST * .85
    PCOST = PLAN A: P$ = "A": GOSUB DisplayPlan
END IF
IF HANDICAP$ = "YES" THEN
    PLAN B = COST * .9
    PCOST = PLAN B: P$ = "B": GOSUB DisplayPlan
END IF
IF (TOAREA$ = "407") AND DIFFAREA AND (CALLLEN < 3.5) THEN
    PLAN C = COST * .8775
    PCOST = PLAN C: 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 PLAN A < PLAN B AND PLAN A < PLAN C THEN PRINT "A": END
    IF PLAN B < PLAN A AND PLAN B < PLAN C 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$ = "3141592653589793238462643383279502884"
PI2$ = "1971693993751058209749445923078164062"
PI3$ = "8620899862803482534211706798214808651"
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
```