```
{ -- FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '81 }
{ -- PASCAL PROGRAM SOLUTIONS }


{1.1}
program One1T81;
{ -- This program will compute percent of heads and tails. }
  var
    H, T, S: Integer;

begin
  Write ('Enter number of heads: ');  Readln (H);
  Write ('Enter number of tails: ');  Readln (T);
  S := T + H;
  Writeln ('PERCENT HEADS: ', H/S * 100 :3:1);
  Writeln ('PERCENT TAILS: ', T/S * 100 :3:1);
end.


{1.2}
program One2T81;
{ -- This program will display the angle of a polygon. }
  var
    N: Integer;

begin
  Write ('Enter number of sides: ');  Readln (N);
  Writeln ('ANGLE=', 180 * (N - 2) div N);
end.


{1.3}
program One3T81;
{ -- This program will compute the value of a function. }
  var
    A, B, C, X: Integer;

begin
  Write ('A, B, C, X: ');  Readln (A, B, C, X);
  Writeln ('AX^2 + BX + C = ', A*X*X + B*X + C);
end.
```

```
{1.4}
program One4T81;
{ -- This program will compute the net price after discounts. }
  var
    I:    Byte;
    P, D: Real;

begin
  Write ('Enter original price: $');  Readln (P);
  for I := 1 to 2 do begin
    Write ('Enter discount ', I, ' percent: ');  Readln (D);
    P := P - P * D / 100
  end;
  Writeln ('FINAL NET PRICE: $', P :3:2);
end.



{1.5}
program One5T81;
{ -- This program will determine the quadrant of a point. }
  var
    X, Y: Integer;

begin
  Write ('Enter X, Y: ');  Readln (X, Y);
  if (X > 0) and (Y > 0) then Writeln ('QUADRANT: I');
  if (X < 0) and (Y > 0) then Writeln ('QUADRANT: II');
  if (X < 0) and (Y < 0) then Writeln ('QUADRANT: III');
  if (X > 0) and (Y < 0) then Writeln ('QUADRANT: IV');
  if X = 0 then Writeln ('LIES ON THE Y-AXIS');
  if Y = 0 then Writeln ('LIES ON THE X-AXIS');
end.
```

```
{2.1}
program Two1T81;
{ -- This program will sum two fractions. }
  var
    A, B, C, D, Num, Den, I: Integer;

begin
  Write ('Enter a, b, c, d: ');  Readln (A, B, C, D);
  Num := A * D + B * C;
  Den := B * D;
    I := Num;
  while (Num mod I <> 0) or (Den mod I <> 0) do
    Dec(I);
  Writeln (Num div I, '/', Den div I);
end.


{2.2}
program Two2T81;
{ -- This program will determine if quad is equilateral. }
  var
    I, Asq, Bsq: Integer;
    A, B:        Array [1..5] of Integer;
    C:           Array [1..5] of Real;

begin
  for I := 1 to 4 do begin
    Write ('Enter point ', I, ': ');  Readln (A[I], B[I]);
  end;
  A[5] := A[1];  B[5] := B[1];
  for I := 1 to 4 do begin
    Asq := (A[I] - A[I+1]) * (A[I] - A[I+1]);
    Bsq := (B[I] - B[I+1]) * (B[I] - B[I+1]);
    C[I] := Sqrt (Asq + Bsq);
  end;
  Write ('QUAD IS ');
  for I := 1 to 3 do
    if Abs (C[I] - C[I+1]) > 0.1 then begin
      Writeln ('NOT EQUILATERAL');  Exit;
    end;
  Writeln ('EQUILATERAL');
end.
```

```
{2.3}
program Two3T81;
{ -- This program will print discount rate for phone call. }
  var
    D, T: Integer;

begin
  Write ('Enter day, time: ');  Readln (D, T);
  if (T >= 1700) and (T < 2300) then Writeln ('20%') else
  if (T >= 2300) or  (T < 700)  then Writeln ('40%') else
  if D = 7 then Writeln ('20%') else
  if D = 1 then Writeln ('40%') else
  Writeln ('NO DISCOUNT');
end.


{2.4}
program Two4T81;
{ -- This program will determine if graph is parallel. }
  var
    A, B, C, D, E, F: Integer;

begin
  Write ('Enter A, B, C: ');  Readln (A, B, C);
  Write ('Enter D, E, F: ');  Readln (D, E, F);
  Write ('LINES ARE ');
  if A * E <> D * B then Write ('NOT ');
  Writeln ('PARALLEL');
end.



{2.5}
program Two5T81;
{ -- This program will find the LCM of 3 integers. }
  var
    A, B, C, S: Integer;

begin
  Write ('Enter three integers: ');
  Readln (A, B, C);  S := 0;
  repeat
    S := S + A;
  until (s mod B = 0) and (S mod C = 0);
  Writeln (S);
end.
```

```
{3.1}
program Thr1T81;
{ -- This program will convert a number from base 10 to B. }
  var
    N, B, J, I, X, Pow: Integer;

begin
  Write ('Enter numeral, base: ');  Readln (N, B);
  J := Trunc (Ln(N) / Ln(B));  Pow := 1;
  for I := 0 to J do  Pow := Pow * B;
  for I := J downto 0 do begin
    Pow := Pow div B;
    X := N div Pow;  Write (X);
    N := N - X * Pow;
  end;
  Writeln;
end.


{3.2}
program Thr2T81;
{ -- This program will print the mode in a list. }
  var
    N, I, J, Max: Integer;
    A, B:         Array [1..20] of Integer;

begin
  Write ('Enter how many numbers: ');  Readln (N);
  for I := 1 to N do begin
    Write ('Enter #: ');  Readln (A[I]);
  end;
  for I := 1 to N do begin
    B[I] := 1;
    for J := I+1 to N do
      if A[I] = A[J] then Inc(B[I]);
    if B[I] > Max then Max := B[I];
  end;
  Write ('MODE(S): ');
  for I := 1 to N do
    if B[I] = Max then Write (A[I], '  ');
  Writeln;
  Writeln ('NUMBER OF OCCURRENCES: ', Max);
end.
```

```
{3.3}
program Thr3T81;
{ -- This program will compute gross weekly pay. }
  var
    E:      String[12];
    R, Pay: Real;
    I:      Byte;
    H:      Array[1..5] of Real;

begin
  Write ('Employee Number: ');  Readln (E);
  Write ('Regular rate of pay/hour: $');  Readln (R);
  Write ('Enter hours for M,T,W,R,F: ');
  Readln (H[1], H[2], H[3], H[4], H[5]);
  Pay := 0;
  for I := 1 to 5 do
    if H[I] <= 8 then
      Pay := Pay + H[I] * R
    else
      Pay := Pay + 8 * R + (H[I]-8) * R * 2;
  Writeln ('EMPLOYEE NUMBER: ', E);
  Writeln ('GROSS WEEKLY PAY: $', Pay :3:2);
end.



{3.4}
program Thr4T81;
{ -- This program will play tic-tac-toe with a user. }
uses Crt;
  const
    Winsq: Array [1..8,1..3] of Integer =
     { -- Board numbering system }
     ((1,2,3), (8,9,4), (7,6,5),
     { -- Sets of 3 winning squares (in addition to above) }
      (1,8,7), (2,9,6), (3,4,5), (1,9,5), (3,9,7));
    { -- Vertical and horizontal coordinates for squares }
    Row: Array [1..9] of Integer = (1,1,1,3,5,5,5,3,3);
    Col: Array [1..9] of Integer = (1,5,9,9,9,5,1,1,5);
    Pl:  Array [0..1] of String[8] = ('YOU', 'COMPUTER');
    Ast = '  |   |';
    Bst = '---------';
  var
    I, Mov, N, P, X:   Integer;
    A: Array [1..9] of Integer;
```

```
function SomeOneWon: Boolean;
{ -- This procedure checks 8 columns, rows, and diagonals. }
begin
  I := 1;  SomeOneWon := False;
  repeat
    if (A[ Winsq[I,1] ] = P) and (A [Winsq[I,2] ] = P) and
       (A[ Winsq[I,3] ] = P) then
    begin
      GotoXY (3, 10);  Writeln (Pl[P], ' WON!');
      SomeOneWon := True;  I := 8;
    end;
    Inc(I);
  until (I = 9);
end;

begin
  ClrScr;  Writeln (Ast);  Writeln (Bst);  Writeln (Ast);
  Writeln (Bst);  Writeln (Ast);
  for I := 1 to 9 do begin
    A[I] := 9;  GotoXY (Col[I], Row[I]);  Write (I);
  end;
  A[9] := 1;  GotoXY (Col[9], Row[9]);  Write ('X');

  for Mov := 2 to 9 do begin
    if Mov in [2, 4, 6, 8] then P := 0 else P := 1;
    if P = 0 then
      begin          { -- Your move }
        repeat
          GotoXY (3, 8);  Write ('Enter # ');  Readln (N);
          GotoXY (10, 8); Write ('  ');
        until (A[N] <> 0) and (A[N] <> 1);
        A[N] := 0;  GotoXY (Col[N], Row[N]);  Write('O');
        if SomeOneWon then Exit;
      end
    else
      begin          { -- Computer's move }
        repeat
          X := Random (9);
        until (A[X] <> 0) and (A[X] <> 1);
        A[X] := 1;  GotoXY (Col[X], Row[X]);  Write('X');
        if SomeOneWon then Exit;
      end
  end;  { -- for Mov }
  GotoXY (3, 10);  Writeln ('TIE GAME');
end.
```

```
{3.5}
program Thr5T81;
{ -- This program will print a list of people who will retire. }
  const
    TM = 4;  TY = 1981;  { -- Todays month/year }
  var
    I, J, N, Y, Yr, X: Integer;
    Xst:               String[40];
    S, BM, Nam: Array[1..9] of String[18];
    BY:         Array[1..9] of Integer;
    A:          Array[1..9] of Integer;
    B:          Array[1..5,1..9] of Integer;
    SandNam:    Array[1..5,1..9] of String[40];

begin
  Write ('Enter number of employees: ');  Readln (N);
  for I := 1 to N do begin
    Writeln;
    Write ('Social Security No.: ');  Readln (S[I]);
    Write ('Name: ');  Readln (Nam[I]);
    Write ('Birthdate: (Month and day): ');  Readln (BM[I]);
    Write ('Birthdate: (Year): ');  Readln (BY[I]);
  end;
         { -- Determine who retires when }
  for Y := TY - 69 to Ty - 65 do begin
    Yr := Y - (TY - 70);  A[Yr] := 0;
    for I := 1 to N do
      if BY[I] <= Y then begin
        Inc(A[Yr]);
        SandNam[Yr, A[Yr]] := '  #' + S[I] + ' ' + Nam[I];
        B[Yr, A[Yr]] := BY[I];
      end;
  end;
         { -- Dispay retirers }
  for Y := 1 to 5 do
    if A[Y] > 0 then begin { -- sort people by birthdates }
      for I := 1 to A[Y]-1 do
        for J := I+1 to A[Y] do
          if B[Y,I] > B[Y,J] then begin
            X := B[Y,I];  B[Y,I] := B[Y,J];  B[Y,J] := X;
            Xst := SandNam[Y,I];
            SandNam[Y,I] := SandNam[Y, J];
            SandNam[Y,J] := Xst;
          end;
      { -- Display retirers in order of dates }
      Writeln;
      Writeln ('RETIRE WITHIN ', Y, ' YEARS');
      for I := 1 to A[Y] do
        Writeln (SandNam[Y, I]);
    end;
end.
```