```
{ -- FLORIDA HIGH SCHOOLS COMPUTING COMPETITION '83 }
{ -- PASCAL PROGRAM SOLUTIONS }


{1.1}
program One1T83;
{ -- This program will round a number to nearest whole number. }
  var
    Num: Real;

begin
  Write ('Enter number: ');  Readln (Num);
  Writeln (Round(Num));
end.


{1.2}
program One2T83;
{ -- This program will display 5 numbers in descending order. }
  var
    I, J, X: Integer;
    A:       Array [1..5] of Integer;

begin
  for I := 1 to 5 do begin
    Write ('Enter number: ');  Readln (A[I]);
  end;
  for I := 1 to 4 do
    for J := I+1 to 5 do
      if A[I] < A[J] then begin
        X := A[I];  A[I] := A[J];  A[J] := X;
      end;

  for I := 1 to 5 do
    Writeln (A[I]);
end.


{1.3}
program One3T83;
{ -- This program will print the factors of a given number. }
  var
    Num, I: Integer;

begin
  Write ('Enter number: ');  Readln (Num);
  for I := 1 to Num do
    if Num mod I = 0 then
      Writeln (I);
end.
```

```
{1.4}
program One4T83;
{ -- This program will produce a birthday card w/name centered. }
  var
    I, J, L, Sp: Integer;
    Name:        String[10];

begin
  Write ('Enter name: ');  Readln (Name);
  for I := 1 to 5 do begin
    Writeln;
    if I in [1, 5] then
      for J := 1 to 12 do
        Write ('*')
    else if (I = 2) then
      Write ('*   HAPPY  *')
    else if (I = 3) then
      Write ('* BIRTHDAY *')
    else begin
      Write ('*');
      L := Length(Name);
      Sp := (11-L) div 2;
      Write (' ': Sp, Name, ' ': 10-L-Sp, '*');
    end;
  end;
end.
```

```
{1.6}
program One6T83;
{ -- This program will print a B for A, C for B, ... Z for A. }
  var
    Ch: Char;

begin
  Write ('Enter Character: ');  Readln (Ch);
  if Ch < 'Z' then
    Writeln (Char(Ord(Ch) + 1))
  else  { -- Z was entered }
    Writeln ('A');
end.
```

```
{1.5}
program One5T83;
{ -- This program will print a ? in random locations. }
uses Crt;
  var
    I, X, Y: Byte;

begin
  ClrScr;  Randomize;
  for I := 1 to 6 do begin
    X := Random(80) + 1;  Y := Random(24) + 1;
    GotoXY (X, Y);  Write ('?');
    Delay (4000);
  end;
end.
```

```
{1.7}
program One7T83;
{ -- This program will print 4 distinct rectangles in corners. }
uses Crt;

procedure Rectangle ({At} Row, Col: Integer);
{ -- This procedure will produce a 10 by 4 rectangle at X, Y }
  var
    I, J: Byte;

begin
  for I := Row to Row+3 do
    if (I = Row) or (I = Row+3) then begin
      GotoXY (Col, I);
      for J := 1 to 10 do
        Write ('*');
      end
    else begin
      GotoXY (Col, I);   Write ('*');
      GotoXY (Col+9, I); Write ('*');
    end;
end;

begin
  ClrScr;
  Rectangle (1, 1);
  Rectangle (1, 65);
  Rectangle (19, 1);
  Rectangle (19, 65);
end.
```

```
{1.8}
program One8T83;
{ -- This program will count the number of e's in a sentence. }
  var
    Sent: String[80];
    I, E: Byte;
    Ch:   Char;

begin
  Write ('Enter sentence: ');  Readln (Sent);
  E := 0;
  for I := 1 to Length(Sent) do begin
    Ch := UpCase( Sent[I] );
    if Ch = 'E' then Inc(E);
  end;
  Writeln (E, ' E''s');
end.
```

```
{1.9}
program One9T83;
{ -- This program will calculate the average score for a person.}
  const
    Name:   Array [1..3] of String[4] = ('JOHN', 'BILL', 'MARY');
    Scores: Array [1..3,1..3] of Byte =
      ((20, 70, 32),  (71, 40, 30),  (80, 42, 73));
  var
    I, J, Total, Ind: Byte;
    St:               String[4];

begin
  Write ('Enter name: ');  Readln (St);
  for I := 1 to 3 do
    if St = Name[I] then Ind := I;
  Total := 0;
  for J := 1 to 3 do
    Total := Total + Scores[Ind, J];
  Writeln ('Average = ', Total / 3 :3:2);
end.
```

```
{1.10}
program One10T83;
{ -- This program will reverse the digits of a 4 digit number. }
  var
    I:  Byte;
    St: String[4];

begin
  Write ('Enter number: ');  Readln (St);
  for I := 4 downto 1 do
    Write (Copy(St, I, 1));
  Writeln;
end.
```

```
{2.1}
program Two1T83;
{ -- This program will calculate the area of a regular hexagon. }
  var
    Perim, S: Real;

begin
  Write ('Enter perimeter: ');  Readln (Perim);
  S := Perim / 6;
  Writeln ( (Sqrt(3)*S/2 * S/2) * 6 :7:4);
end.
```

```
{2.2}
program Two2T83;
{ -- This program will convert a base 8 num to a base 2 num. }
  var
    I, Digit:  Byte;
    FirstDig:  Byte;
    Ch:        Char;
    Num:       String[4];
    St:        String[12];

begin
  Write ('Enter number: ');  Readln (Num);
  St := '';
  for I := 1 to Length(Num) do begin
    Ch := Num[I];
    Digit := Ord(Ch) - Ord('0');
    case Digit of
      0: St := St + '000';
      1: St := St + '001';
      2: St := St + '010';
      3: St := St + '011';
      4: St := St + '100';
      5: St := St + '101';
      6: St := St + '110';
      7: St := St + '111';
    end;
  end;
  FirstDig := 1;
  while Copy(St, FirstDig, 1) = '0' do
    Inc(FirstDig);
  Writeln (Copy(St, FirstDig, Length(St)-FirstDig+1));
end.
```

```
{2.3}
program Two3T83;
{ -- This program will add several items with tax (5%). }
  var
    Item, Tax, Total: Real;

begin
  Total := 0;
  Write ('Enter item: ');  Readln (Item);
  while Item <> -999 do begin
    Total := Total + Item;
    Write ('Enter item: ');  Readln (Item);
  end;
  Writeln ('SUBTOTAL = $', Total: 5:2);
  Tax := Total * 0.05;
  Writeln ('TAX      = $', Tax:   5:2);
  Total := Total + Tax;
  Writeln ('TOTAL    = $', Total: 5:2);
end.
```

```
{2.4}
program Two4T83;
{ -- This program will divide the screen into 4 rectangles. }
uses Crt;
  var
    Ch:   Char;
    I, J: Integer;

begin
  Write ('Enter character: ');  Readln (Ch);
  ClrScr;
  for I := 1 to 24 do
    if I <> 12 then
      Writeln (' ': 39, Ch)
    else
      for J := 1 to 80 do
        Write (Ch);
end.
```

```
{2.5}
program Two5T83;
{ -- This program will print the greatest and least in a set. }
  var
    Max, Min, Num: Real;

begin
  Max := -900;  Min := 900;
  Write ('Enter number: ');  Readln (Num);
  while Num <> -999 do begin
    if Num < Min then Min := Num
    else if Num > Max then Max := Num;
    Write ('Enter number: ');  Readln (Num);
  end;
  Writeln ('GREATEST = ', Max :5:2);
  Writeln ('LEAST = ', Min :5:2);
end.
```

```
{2.6}
program Two6T83;
{ -- This program will print the sum, mean, median. }
  var
    I, J:   Byte;
    Sum, X: Real;
    A:      Array [1..10] of Real;

begin
  Sum := 0;
  for I := 1 to 10 do begin
    Write ('Enter number: ');  Readln (A[I]);
    Sum := Sum + A[I];
  end;
  { -- Sort 10 numbers }
  for I := 1 to 9 do
    for J := I+1 to 10 do
      if A[I] > A[J] then begin
        X := A[I];  A[I] := A[J];  A[J] := X;
      end;

  Writeln ('SUM = ', Sum: 5:2);
  Writeln ('MEAN = ', Sum / 10 :5:2);
  Writeln ('MEDIAN = ', (A[5] + A[6])/2 :5:2);
end.
```

```
{2.7}
program Two7T83;
{ -- This program will reverse the words in a sentence. }
{ -- Assume 1 space between each word. }
  var
    Sent:   String[80];
    Word:   Array [1..10] of String[10];
    I, Num: Byte;
    Ch:     Char;

begin
  Write ('Enter sentence: ');  Readln (Sent);
  Num := 1;  Word[Num] := '';
  for I := 1 to Length(Sent) do begin
    Ch := Sent[I];
    if Ch <> ' ' then
      Word[Num] := Word[Num] + Ch
    else begin
      Inc(Num);
      Word[Num] := '';
    end;
  end;
  for I := Num downto 1 do
    Write (Word[I], ' ');
  Writeln;
end.
```

```
{2.8}
program Two8T83;
{ -- This program will convert cubic feet to cubic meters. }
{ -- (1 in. = 2.54 cm) }
  var
    CF, CM, CM3: Real;

begin
  Write ('Enter cubic feet: ');  Readln (CF);
  CM3 := CF * (12 * 2.54) * (12 * 2.54) * (12 * 2.54);
  CM := CM3/ 100 / 100 / 100;
  Writeln (CM :7:4, ' CUBIC METERS');
end.
```

```
{2.9}
program Two9T83;
{ -- This program will find sum of Ys for Xs for Y=2(X+5). }
  var
    A, B, X, Sum: Integer;

begin
  Write ('Enter a and b: ');  Readln (A, B);  Sum := 0;
  for X := A to B do
    Sum := Sum + 2 * (X+5);
  Writeln ('SUM = ', Sum);
end.
```

```
{2.10}
program Two10T83;
{ -- This program will print 1 char. for 10 sec, 2 for 10 sec.. }
uses Crt;
  var
    I, J: Byte;
    Ch:   Char;

begin
  Write ('Enter character: ');  Readln (Ch);  ClrScr;
  for I := 1 to 10 do begin
    for J := 1 to I do
      Write (Ch);
    Delay (5000);
    ClrScr;  Delay (500);
  end;
end.
```

```
{3.1}
program Thr1T83;
{ -- This program converts a number for one base to another. }
  var
    Base1, Base2, Num1V, Num2, Power: Integer;
    I, J, K, X, Digit:                Integer;
    Num1:                             String[7];
    Ch:                               Char;

begin
  Write ('ENTER NUMBER: ');  Readln (Num1);
  Write ('ENTER BASE: ');    Readln (Base1);
  Write ('CONVERT TO BASE: '); Readln (Base2);
  Write ('ANSWER IS ');

  { -- Convert Num1 to base 10 number Num1V }
  Num1V := 0;
  for I := 1 to Length(Num1) do begin
    Ch := Num1[I];
    Digit := Ord(Ch) - Ord('0');
    Power := 1;
    for J := 1 to Length(Num1) - I do
      Power := Power * Base1;
    Num1V := Num1V + Digit * Power;
  end;

  { -- Convert Num1V to Base2 number }
  J := Trunc(Ln(Num1V) / Ln(Base2));
  for I := J downto 0 do begin
    Power := 1;
    for K := 1 to I do  Power := Power * Base2;
    X := Num1V div Power;
    Write (X);
    Num1V := Num1V - X * Power;
  end;
  Writeln;
end.
```

```
{3.2}
program Thr2T83;
{ -- This program determines what triangle is made w/3 points. }
  var
    X1, Y1, X2, Y2, X3, Y3: Integer;
    D1, D2, D3:             Real;

begin
  Write ('Enter X1, Y1: ');  Readln (X1, Y1);
  Write ('Enter X2, Y2: ');  Readln (X2, Y2);
  Write ('Enter X3, Y3: ');  Readln (X3, Y3);

  { -- Calculate distances }
  D1 := Sqrt ((X1-X2)*(X1-X2) + (Y1-Y2)*(Y1-Y2));
  D2 := Sqrt ((X2-X3)*(X2-X3) + (Y2-Y3)*(Y2-Y3));
  D3 := Sqrt ((X3-X1)*(X3-X1) + (Y3-Y1)*(Y3-Y1));

  { -- No triangle can be formed if sum of 2 sides equals third. }
  if (D1+D2 = D3) or (D1+D3 = D2) or (D2+D3 = D1) then
    Writeln ('NOT A TRIANGLE')
  else if (D1 = D2) and (D2 = D3) then
    Writeln ('EQUILATERAL')
  else if (D1 = D2) or (D2 = D3) or (D1 = D3) then
    Writeln ('ISOSCELES')
  else
    Writeln ('SCALENE');
end.
```

```
{3.3}
program Thr3T83;
{ -- This program randomly selects an X, Y in 10 x 10 grid. User
  -- guesses numbers; if guess is wrong, a direction is given. }

  var
    X, Y, A, B: Byte;

begin
  Randomize;
  X := Random(10) + 1;  Y := Random(10) + 1;
  repeat
    Write ('Enter X, Y: ');  Readln (A, B);
         if (A = X) and (B < Y) then Writeln ('UP')
    else if (A = X) and (B > Y) then Writeln ('DOWN')
    else if (A > X) and (B = Y) then Writeln ('LEFT')
    else if (A < X) and (B = Y) then Writeln ('RIGHT')
    else if (A < X) and (B < Y) then Writeln ('UP AND RIGHT')
    else if (A < X) and (B > Y) then Writeln ('DOWN AND RIGHT')
    else if (A > X) and (B < Y) then Writeln ('UP AND LEFT')
    else if (A > X) and (B > Y) then Writeln ('DOWN AND LEFT');
  until (A=X) and (B=Y);
end.
```

```
{3.4}
program Thr4T83;
{ -- This program will divide 1st number by 2nd out to N places. }
  var
    Num1, Num2, Places, I, X: Integer;

begin
  Write ('ENTER FIRST NUMBER: ');  Readln (Num1);
  Write ('ENTER SECOND NUMBER: '); Readln (Num2);
  Write ('ENTER NUMBER OF DECIMAL PLACES: ');  Readln (Places);
  Write ('QUOTIENT IS ');
  X := Num1 div Num2;  Write (X, '.');
  Num1 := Num1 - Num2*X;
  for I := 1 to Places do begin
    Num1 := Num1 * 10;
    X := Num1 div Num2;
    Write (X);
    Num1 := Num1 - Num2*X;
  end;
end.
```

```
 {3.5}
program Thr5T83;
{ -- 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). }
uses Crt;
  var
    I, J, X, R1, R2, IndX, IndY: Byte;
    Digit, BlankX, BlankY:       Byte;
    A:                           Array [1..3, 1..3] of Byte;
    Valid:                       Boolean;
    DigSt:                       String[1];
    Code:                        Integer;
```

```
begin
  { -- Randomly place numbers in Array A. }
  Randomize;
  for I := 1 to 3 do
    for J := 1 to 3 do
      A[I,J] := (I-1)*3 + J-1;
  for I := 1 to 3 do
    for J := 1 to 3 do begin  { -- swap array values }
      R1 := Random(3) + 1;  R2 := Random(3) + 1;
      X := A[I,J];  A[I,J] := A[R1,R2];  A[R1,R2] := X;
    end;
  repeat
    { -- Display array }
    ClrScr;
    for I := 1 to 3 do begin
      for J := 1 to 3 do
        if A[I,J] > 0 then Write (A[I,J], '  ')
        else begin
          Write ('   ');
          BlankX := I;  BlankY := J;
        end;
      Writeln;
    end;

    { -- Accept valid digit or 9 }
    Valid := False;
    repeat
      DigSt := ''; while DigSt = '' do DigSt := ReadKey;
      Val(DigSt,Digit,Code);
      for I := 1 to 3 do
        for J := 1 to 3 do
          if Digit = A[I,J] then begin
            IndX := I;  IndY := J;
          end;
      if Abs(BlankX - IndX) + Abs(BlankY - IndY) = 1 then
        { -- adjacent }
        Valid := True;
    until Valid or (Digit = 9);

    if Valid then begin  { -- move digit in space }
      X := A[IndX,IndY];  A[IndX,IndY] := A[BlankX,BlankY];
      A[BlankX,BlankY] := X;
    end;
  until Digit = 9;  { -- 9 pressed }
end.
```

```
{3.6}
program Thr6T83;
{ -- This program will store a list of words and provide options.}
  var
    Option, I, J, Num: Byte;
    Word:              Array [1..10] of String[10];
    DeleteW:           String[10];

begin
  Num := 0;
  repeat
    Writeln;
    Writeln ('1. ADD A WORD TO THE LIST');
    Writeln ('2. DELETE A WORD FROM THE LIST');
    Writeln ('3. DISPLAY THE ENTIRE LIST');
    Readln (Option);
    case Option of
      1: begin
           Inc(Num);
           Write ('Enter word: ');  Readln (Word[Num]);
         end;
      2: begin
           Write ('Enter word: ');  Readln (DeleteW);
           I := 1;
           while (I <= Num) and (Word[I] <> DeleteW) do
             Inc(I);
           for J := I to Num-1 do Word[J] := Word[J+1];
           Dec(Num);
         end;
      3: for I := 1 to Num do
           Writeln (Word[I]);
    end;
  until Option > 3;
end.
```

```
{3.7}
program Thr7T83;
{  --  This  program  will  solve  cryptorithms  with  two  2-letter
addends
  -- and a 3-letter sum, using only the letters A,B,C,D, and E. }

  var
    St1, St2, St3:       String[3];
    Letters, Numbers:    String[7];
    FirstLet, UniqueLet: Array [1..7] of Integer;
    N1St, N2St, SumSt:   String[3];
    Ch:                  Char;
    Solution, AtLeast1:  Boolean;
    I, J, N1, N2, Sum, NumLet: Integer;
```

```
begin
  Write ('Enter FIRST ADDEND: ');  Readln (St1);
  Write ('Enter SECOND ADDEND: '); Readln (St2);
  Write ('Enter SUM: ');           Readln (St3);
  Letters := St1 + St2 + St3;  NumLet := 0;  AtLeast1 := False;

  { Put in FirstLet[] the index of the first occurence of letter }
  for I := 1 to 7 do begin
    Ch := Letters[I];
    FirstLet[I] := Pos(Ch, Letters);
    if FirstLet[I] = I then begin { -- This is a new letter. }
      Inc(NumLet);
      UniqueLet[NumLet] := I;
    end;
  end;

  for N1 := 10 to 98 do             { -- N1 must be 2 digits, >9 }
    for N2 := 100-N1 to 98 do begin { -- N2 must be 2 digits, >9 }
      Sum := N1 + N2;               { -- Sum must be 3 digits,>99}
      Str (N1, N1St);  Str (N2, N2St);  Str (Sum, SumSt);
      Numbers := N1St + N2St + SumSt;
      I := 1;  Solution := True;
      { -- Check if similar letters correspond to similar numbers}
      repeat
        Ch := Numbers[I];
        if Ch <> Copy (Numbers, FirstLet[I], 1) then
          Solution := False;
        Inc(I);
      until (I > 7) or not Solution;

      { -- Check if unique letters correspond to unique digits }
      for I := 1 to NumLet-1 do
        for J := I+1 to NumLet do
          if Numbers[UniqueLet[I]] = Numbers[UniqueLet[J]] then
            Solution := False;

      if Solution then begin  { -- Display solution }
        for I := 1 to NumLet do begin
          Write (Letters[UniqueLet[I]], ' = ');
          Writeln (Numbers[UniqueLet[I]]);
        end;  Exit;
        Writeln;  AtLeast1 := True;
      end;
    end;  { -  for N2 }
    if not AtLeast1 then
      Writeln ('NO SOLUTION POSSIBLE');
end.
```

```
{3.8}
program Thr8T83;
{ -- This program will simulate random frog jumps on nine pads. }
uses Crt;
  var
    I, F, Num: Byte;

begin
  Randomize;  ClrScr;
  for I := 1 to 10 do begin
    GotoXY (1, 1);  ClrEol;
    GotoXY (1, 2);  Writeln ('- - - - - - - - -');
    F := 5;
    GotoXY (F*2-1, 1);  Write ('F');  Num := 0;
    repeat
      if Random(2) = 1 then { -- go right }
        Inc(F)
      else  { -- go left }
        Dec(F);
      GotoXY (1, 1);      ClrEol;
      GotoXY (F*2-1, 1);  Write ('F');  Delay (50);
      Inc(Num);
    until (F = 1) or (F = 9);
    GotoXY (I*3, 5);  Write (Num);
  end;
end.
```

```
{3.9}
program Thr9T83;
{ -- This program will allow a user to position a cursor under a
  -- sentence using the L and R keys.  Space bar deletes letter. }
uses Crt;

  var
    I, Col: Byte;
    Sent:   String[80];
    Ch:     Char;

begin
  ClrScr;
  Write ('Enter Sentence: ');  Readln (Sent);  { -- Starts at 17 }
  Col := 17;
  repeat
    GotoXY (Col, 2);
    repeat
      Ch := ReadKey;  Ch := UpCase(Ch);
    until (Ch in ['R', 'L', ' ']) or (Ch = Chr(27));
    if Ch = 'R' then       { -- move cursor to right }
      Inc(Col)
    else if Ch = 'L' then  { -- move cursor to left }
      Dec(Col)
    else if Ch = ' ' then  { -- delete character above cursor }
      Delete (Sent, Col-16, 1);
    GotoXY (17, 1);  Writeln (Sent, ' ');
  until (Ch = Chr(27)) or (Length(Sent) = 1);
end.
```

```
{3.10}
program Thr10T83;
{ -- This program will simulate the movement of a pool ball on a
  -- rectangular pool table.  It moves in a 45 degree angle. }
uses Crt, Graph3;

  var
    W, L, WI, LI, I, X, Y, XDir, YDir: Integer;
    Finished: Boolean;

begin
  Write ('Enter Width, Length: ');  Readln (W, L);
  ClrScr;  GraphMode;
  WI := 10;  LI := 10;
  { -- Draw Pool Table }
  for I := 0 to W do
    Draw (0,I*WI, L*LI,I*WI, 1);
  for I := 0 to L do
    Draw (I*LI,0, I*LI,W*WI, 1);

  X := 0;  Y := W*WI;  XDir := 1;  YDir := -1;
  repeat
    Plot (X, Y, 0);
    X := X + XDir;  Y := Y + YDir;
    Plot (X, Y, 2);  Delay (10);
    if (X = 0) or (X = L*LI) then
      XDir := -1 * XDir;
    if (Y = 0) or (Y = W*WI) then
      YDir := -1 * YDir;

    Finished := True;  GotoXY (1,20);
    if (X = 0) and (Y = 0) then
      Writeln ('LEFT-TOP')
    else if (X = 0) and (Y = W*WI) then
      Writeln ('LEFT-BOTTOM')
    else if (X = L*LI) and (Y = 0) then
      Writeln ('RIGHT-TOP')
    else if (X = L*LI) and (Y = W*WI) then
      Writeln ('RIGHT-BOTTOM')
    else
      Finished := False;
  until Finished;
end.
```