Ejemplos de programas de Datos Estructurados

4. Programa que ordena hasta cien números enteros introducidos vía teclado utilizando el método de la Burbuja (*Bubble Sort*)

```
(*********************************************************)
(* Programa burbuja                                      *)
(* Ordena una sucesion de numeros enteros introducidos   *)
(* por teclado                                           *)
(* Division de Informatica Ind. ETSI Industriales. UPM   *)
(*********************************************************)
program burbuja;
const n_max = 100;
var i,j,n,w : integer;
          v : array[1..n_max] of integer;
begin
writeln('  Cuantos numeros quiere ordenar?  ');
readln(n);
writeln('  Introduzca los numeros:  ');
for i:=1 to n do readln(v[i]);
for i:=1 to n-1 do
      for j:=1 to n-i do
            if v[j]>v[j+1] then begin
                           w:=v[j];
                           v[j]:=v[j+1];
                                   v[j+1]:=w
                                   end;
writeln;
writeln('  Los numeros ordenados son:  ');
for i:=1 to n do writeln(v[i])
end.
```

5. Programa para calcular el valor de un polinomio p de grado n en un punto x. Dado el polinomio de grado n:

$$P_n(x) = a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + a_{n-2} \cdot x^{n-2} + \ldots + a_2 \cdot x^2 + a_1 \cdot x + a_0$$

se desea calcular el valor del polinomio en un punto. Esta expresión del polinomio presenta un pequeño inconveniente: en TurboPascal no hay ninguna función predefinida (estándar) que calcule y devuelva la potencia n-ésima de un número. Una solución sería la de manipular la expresión del polinomio para que quede definida únicamente mediante sumas y multiplicaciones, que son operadores definidos en TurboPascal. Agrupando términos en la expresión anterior de la siguiente manera...

$$P_n(x) = \left(a_n \cdot x + a_{n-1}\right) \cdot x^{n-1} + a_{n-2} \cdot x^{n-2} + \ldots + a_2 \cdot x^2 + a_1 \cdot x + a_0$$

$$P_n(x) = \left(\left(a_n \cdot x + a_{n-1}\right) \cdot x + a_{n-2}\right) \cdot x^{n-2} + \ldots + a_2 \cdot x^2 + a_1 \cdot x + a_0$$

$$P_n(x) = \left(\left(\left(a_n \cdot x + a_{n-1}\right) \cdot x + a_{n-2}\right) \cdot x + \ldots + a_2\right) \cdot x^2 + a_1 \cdot x + a_0 \qquad \ldots$$

se llega a la expresión

$$P_n(x) = \left(\left(\left(\left(a_n \cdot x + a_{n-1}\right) \cdot x + a_{n-2}\right) \cdot x + \ldots + a_2\right) \cdot x + a_1\right) \cdot x + a_0$$

que es conocida como *regla de Horner*. Este cálculo se reduce a tan sólo dos sentencias en TurboPascal:

```
Poli:=0;
For I:=n DownTo 0 Do Poli:=Poli*x+a[I];
```

que podrían quedar incluidas en un programa de la siguiente manera:

```
Program Polinomio;
Const  Max = 100;
Var     x, Poli    : Real;
```

```
        a         : Array [0..Max]  of Real;
        n, I      : Integer;
Begin
Write('Introduce el grado del polinominio : ');
ReadLn(n);
WriteLn('Introduce los coeficientes del polinomio.');
For I:=0 To n Do
      Begin
      Write('a(',I,') = ');
      ReadLn(a[I])
      End;
Write('Introduce el valor de x : ');
ReadLn(x);
Poli:=0;
For I:=n DownTo 0 Do Poli:=Poli*x+a[I];
WriteLn('El valor del polinomio en x=',x,' es:',Poli)
End.
```