

## Trabajo Práctico N° 4.1: Vectores (Parte 1).

### Ejercicio 1.

*Dado el siguiente programa:*

```
program TP4_E1;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i: integer;
begin
  for i:= 1 to 10 do
    numeros[i]:=i;
  for i:= 2 to 10 do
    numeros[i]:=numeros[i]+numeros[i-1]
end.
```

(a) ¿Qué valores toma la variable *numeros* al finalizar el primer bloque *for*?

(b) Al terminar el programa, ¿con qué valores finaliza la variable *numeros*?

```
program TP4_E1ab;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i: integer;
begin
  for i:= 1 to 10 do
    begin
      numeros[i]:=i;
      if (i<10) then
        write(numeros[i],', ')
      else
        writeln(numeros[i])
    end;
  for i:= 2 to 10 do
    begin
      numeros[i]:=numeros[i]+numeros[i-1];
      if (i<10) then
        write(numeros[i],', ')
      else
        writeln(numeros[i])
    end;
end.
```

Los valores que toma la variable “*numeros*” al finalizar el primer bloque *for* son 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.

Al terminar el programa, la variable “*numeros*” finaliza con los valores 1, 3, 6, 10, 15, 21, 28, 36, 45, 55.

(c) Si se desea cambiar la línea 11 por la sentencia: *for i:=1 to 9 do*, ¿cómo debe modificarse el código para que la variable *numeros* contenga los mismos valores que en (1.b)?

```
program TP4_E1c;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i: integer;
begin
  for i:= 1 to 10 do
  begin
    numeros[i]:=i;
    if (i<10) then
      write(numeros[i],', ')
    else
      writeln(numeros[i])
    end;
  for i:= 1 to 9 do
  begin
    numeros[i+1]:=numeros[i+1]+numeros[i];
    if (i+1<10) then
      write(numeros[i+1],', ')
    else
      writeln(numeros[i+1])
    end;
  end.
end.
```

(d) ¿Qué valores están contenidos en la variable *numeros* si las líneas 11 y 12 se reemplazan por *for i:= 1 to 9 do numeros[i+1]:=numeros[i];*?

```
program TP4_E1d;
{$codepage UTF8}
uses crt;
type
  vnums=array[1..10] of integer;
var
  numeros: vnums;
  i: integer;
begin
  for i:= 1 to 10 do
  begin
    numeros[i]:=i;
    if (i<10) then
      write(numeros[i],', ')
    else
      writeln(numeros[i])
    end;
  for i:= 1 to 9 do
  begin
    numeros[i+1]:=numeros[i];
    if (i<9) then
      write(numeros[i],', ')
    else
      writeln(numeros[i])
    end;
  end.
end.
```

```
    writeln(numeros[i])  
  end;  
end.
```

Los valores que están contenidos en la variable “*numeros*” si las líneas 11 y 12 se reemplazan por *for i:= 1 to 9 do numeros[i+1]:=numeros[i]* son 1, 1, 1, 1, 1, 1, 1, 1, 1, 1.

## Ejercicio 2.

Dado el siguiente programa, completar las líneas indicadas, considerando que:

- El módulo `cargarVector` debe leer números reales y almacenarlos en el vector que se pasa como parámetro. Al finalizar, debe retornar el vector y su dimensión lógica. La lectura finaliza cuando se ingresa el valor 0 (que no debe procesarse) o cuando el vector está completo.
- El módulo `modificarVectorYSumar` debe devolver el vector con todos sus elementos incrementados con el valor `n` y también debe devolver la suma de todos los elementos del vector.

```

program TP4_E2;
{$codepage UTF8}
uses crt;
const
  cant_datos=150;
  num_salida=0;
type
  vdatos=array[1..cant_datos] of real;
procedure cargarVector(var v: vdatos; var dimL: integer);
var
  num: real;
begin
  num:=num_salida+random(101)/10;
  while ((num<>num_salida) and (dimL<cant_datos)) do
  begin
    dimL:=dimL+1;
    v[dimL]:=num;
    num:=num_salida+random(101)/10;
  end;
end;
var
  num: real;
begin
  num:=random(101)/10;
  while ((num<>num_salida) and (dimL<cant_datos)) do
  begin
    dimL:=dimL+1;
    v[dimL]:=num;
    num:=random(101)/10;
  end;
end;
procedure modificarVectorYSumar(var v: vdatos; dimL: integer; n: real; var suma: real);
var
  i: int16;
begin
  for i:= 1 to dimL do
  begin
    v[i]:=v[i]+n;
    suma:=suma+v[i];
  end;
end;
var
  datos: vdatos;
  dim: integer;
  num, suma: real;
begin
  randomize;
  dim:=0; suma:=0;
  cargarVector(datos,dim);

```

```
num:=1+random(10);
modificarVectorYSumar(datos,dim,num,suma);
textcolor(green); write('La suma de los valores es '); textcolor(red); writeln(suma:0:2);
textcolor(green); write('Se procesaron '); textcolor(red); write(dim); textcolor(green);
write(' números');
end.
```

**Ejercicio 3.**

Se dispone de un vector con números enteros, de dimensión física  $\text{dimF}$  y dimensión lógica  $\text{dimL}$ .

(a) Realizar un módulo que imprima el vector desde la primera posición hasta la última.

(b) Realizar un módulo que imprima el vector desde la última posición hasta la primera.

(c) Realizar un módulo que imprima el vector desde la mitad ( $\text{dimL} \text{ DIV } 2$ ) hacia la primera posición y desde la mitad más uno hacia la última posición.

(d) Realizar un módulo que reciba el vector, una posición  $X$  y otra posición  $Y$ , e imprima el vector desde la posición  $X$  hasta la  $Y$ . Asumir que tanto  $X$  como  $Y$  son menores o igual a la dimensión lógica. Y considerar que, dependiendo de los valores de  $X$  e  $Y$ , podría ser necesario recorrer hacia adelante o hacia atrás.

(e) Utilizando el módulo implementado en el inciso anterior, volver a realizar los incisos (a), (b) y (c).

```
program TP4_E3;
{$codepage UTF8}
uses crt;
type
  t_vector_numeros=array of int16;
procedure crear_vector_numeros(var vector_numeros: t_vector_numeros; dimF: int16);
begin
  setLength(vector_numeros,dimF);
end;
procedure cargar_vector_numeros(var vector_numeros: t_vector_numeros; dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    vector_numeros[i]:=random(high(int16));
  end;
end;
procedure imprimir_1adimL(vector_numeros: t_vector_numeros; dimL: int16);
var
  i: int16;
begin
  for i:= 1 to dimL do
    if (i<dimL) then
      write(vector_numeros[i],', ')
    else
      writeln(vector_numeros[i]);
  end;
end;
procedure imprimir_dimLa1(vector_numeros: t_vector_numeros; dimL: int16);
var
  i: int16;
begin
  for i:= dimL downto 1 do
    if (i>1) then
      write(vector_numeros[i],', ')
    else
      writeln(vector_numeros[i]);
  end;
end;
procedure imprimir_dimLdiv2(vector_numeros: t_vector_numeros; dimL: int16);
var
```

```

    i, dimLdiv2, dimLdiv2mas1: int16;
begin
    dimLdiv2:=dimL div 2; dimLdiv2mas1:=dimLdiv2+1;
    for i:= dimLdiv2 downto 1 do
        if (i>1) then
            write(vector_numeros[i],', ')
        else
            writeln(vector_numeros[i]);
        for i:= dimLdiv2mas1 to dimL do
            if (i<dimL) then
                write(vector_numeros[i],', ')
            else
                writeln(vector_numeros[i]);
        end;
    end;
    procedure imprimir_general(vector_numeros: t_vector_numeros; numX, numY: int16);
    var
        i: int16;
    begin
        if (numX<=numY) then
            for i:= numX to numY do
                if (i<numY) then
                    write(vector_numeros[i],', ')
                else
                    writeln(vector_numeros[i])
            end;
        else
            for i:= numX downto numY do
                if (i>numY) then
                    write(vector_numeros[i],', ')
                else
                    writeln(vector_numeros[i]);
            end;
        end;
    var
        vector_numeros: t_vector_numeros;
        dimF, dimL: int16;
    begin
        randomize;
        dimF:=2+random(9);
        dimL:=2+random(dimF-1);
        crear_vector_numeros(vector_numeros,dimF);
        if (dimL>0) then
            begin
                cargar_vector_numeros(vector_numeros,dimL);
                imprimir_1adimL(vector_numeros,dimL);
                imprimir_dimLa1(vector_numeros,dimL);
                imprimir_dimLdiv2(vector_numeros,dimL);
                imprimir_general(vector_numeros,1,dimL);
                imprimir_general(vector_numeros,dimL,1);
                imprimir_general(vector_numeros,dimL div 2,1);
                imprimir_general(vector_numeros,dimL div 2+1,dimL);
            end;
        end.

```

**Ejercicio 4.**

Se dispone de un vector con 100 números enteros. Implementar los siguientes módulos:

(a) *posicion*: dado un número  $X$  y el vector de números, retorna la posición del número  $X$  en dicho vector o el valor -1 en caso de no encontrarse.

(b) *intercambio*: recibe dos valores  $x$  e  $y$  (entre 1 y 100) y el vector de números y retorna el mismo vector, donde se intercambiaron los valores de las posiciones  $x$  e  $y$ .

(c) *sumaVector*: retorna la suma de todos los elementos del vector.

(d) *promedio*: devuelve el valor promedio de los elementos del vector.

(e) *elementoMaximo*: retorna la posición del mayor elemento del vector.

(f) *elementoMinimo*: retorna la posición del menor elemento del vector.

```
program TP4_E4;
{$codepage UTF8}
uses crt;
const
    num_total=100;
type
    t_numero=1..num_total;
    t_vector_numeros=array[t_numero] of int16;
procedure cargar_vector_numeros(var vector_numeros: t_vector_numeros);
var
    i: t_numero;
begin
    for i:= 1 to num_total do
        vector_numeros[i]:=1+random(200);
    end;
function posicion(vector_numeros: t_vector_numeros; numX: int16): int16;
var
    pos: int16;
begin
    pos:=1;
    while ((pos<=num_total) and (vector_numeros[pos]<>numX)) do
        pos:=pos+1;
    if (pos<=num_total) then
        posicion:=pos
    else
        posicion:=-1;
    end;
procedure intercambio(var vector_numeros: t_vector_numeros; numX, numY: int16);
var
    num_aux: int16;
begin
    num_aux:=vector_numeros[numX];
    vector_numeros[numX]:=vector_numeros[numY];
    vector_numeros[numY]:=num_aux;
end;
function sumaVector(vector_numeros: t_vector_numeros): int16;
var
    i: t_numero;
    suma: int16;
begin
    suma:=0;
```



```

    for i:= 1 to num_total do
        suma:=suma+vector_numeros[i];
        sumaVector:=suma;
    end;
function promedio(vector_numeros: t_vector_numeros): real;
begin
    promedio:=sumaVector(vector_numeros)/num_total;
end;
function elementoMaximo(vector_numeros: t_vector_numeros): int16;
var
    i: t_numero;
    ele_max, pos_max: int16;
begin
    ele_max:=low(int16); pos_max:=0;
    for i:= 1 to num_total do
        if (vector_numeros[i]>ele_max) then
            begin
                ele_max:=vector_numeros[i];
                pos_max:=i;
            end;
    elementoMaximo:=pos_max;
end;
function elementoMinimo(vector_numeros: t_vector_numeros): int16;
var
    i: t_numero;
    ele_min, pos_min: int16;
begin
    ele_min:=high(int16); pos_min:=0;
    for i:= 1 to num_total do
        if (vector_numeros[i]<ele_min) then
            begin
                ele_min:=vector_numeros[i];
                pos_min:=i;
            end;
    elementoMinimo:=pos_min;
end;
var
    vector_numeros: t_vector_numeros;
    numX, posX, posY: int16;
begin
    randomize;
    cargar_vector_numeros(vector_numeros);
    numX:=1+random(200);
    textcolor(green); write('La posición del número '); textcolor(red); write(numX);
textcolor(green); write(' en el vector es '); textcolor(red);
writeln(posicion(vector_numeros,numX));
    posX:=1+random(100);
    posY:=1+random(100);
    textcolor(green); write('Pre-intercambio, en las posiciones '); textcolor(red); write(posX);
textcolor(green); write(' y '); textcolor(red); write(posY); textcolor(green); write(' se
tienen los valores '); textcolor(red); write(vector_numeros[posX]); textcolor(green); write('
y '); textcolor(red); write(vector_numeros[posY]); textcolor(green); writeln('
respectivamente');
    intercambio(vector_numeros,posX,posY);
    textcolor(green); write('Post-intercambio, en las posiciones '); textcolor(red);
write(posX); textcolor(green); write(' y '); textcolor(red); write(posY); textcolor(green);
write(' se tienen los valores '); textcolor(red); write(vector_numeros[posX]);
textcolor(green); write(' y '); textcolor(red); write(vector_numeros[posY]); textcolor(green);
writeln(' respectivamente');
    textcolor(green); write('La suma de todos los elementos del vector es '); textcolor(red);
writeln(sumaVector(vector_numeros));
    textcolor(green); write('El valor promedio de los elementos del vector es ');
textcolor(red); writeln(promedio(vector_numeros):0:2);
    textcolor(green); write('La posición del mayor elemento del vector es '); textcolor(red);
writeln(elementoMaximo(vector_numeros));

```

```
textcolor(green); write('La posición del menor elemento del vector es '); textcolor(red);  
write(elementoMinimo(vector_numeros));  
end.
```

## Ejercicio 5.

Utilizando los módulos implementados en el Ejercicio 4, realizar un programa que lea números enteros desde teclado (a lo sumo, 100) y los almacene en un vector. La carga finaliza al leer el número 0. Al finalizar la carga, se debe intercambiar la posición del mayor elemento por la del menor elemento e informar la operación realizada de la siguiente manera: “El elemento máximo ... que se encontraba en la posición ... fue intercambiado con el elemento mínimo ... que se encontraba en la posición ...”.

```

program TP4_E5;
{$codepage UTF8}
uses crt;
const
    num_total=100;
    num_salida=0;
type
    t_numero=1..num_total;
    t_vector_numeros=array[t_numero] of int16;
procedure cargar_vector_numeros(var vector_numeros: t_vector_numeros; var numeros: int16);
var
    num: int16;
begin
    num:=num_salida+random(101);
    while ((num<>num_salida) and (numeros<num_total)) do
    begin
        numeros:=numeros+1;
        vector_numeros[numeros]:=num;
        num:=num_salida+random(101);
    end;
end;
procedure elementoMaximo(vector_numeros: t_vector_numeros; numeros: int16; var ele_max,
pos_max: int16);
var
    i: t_numero;
begin
    for i:= 1 to numeros do
        if (vector_numeros[i]>ele_max) then
        begin
            ele_max:=vector_numeros[i];
            pos_max:=i;
        end;
    end;
end;
procedure elementoMinimo(vector_numeros: t_vector_numeros; numeros: int16; var ele_min,
pos_min: int16);
var
    i: t_numero;
begin
    for i:= 1 to numeros do
        if (vector_numeros[i]<ele_min) then
        begin
            ele_min:=vector_numeros[i];
            pos_min:=i;
        end;
    end;
end;
procedure intercambio(var vector_numeros: t_vector_numeros; pos_max, pos_min: int16);
var
    num_aux: int16;
begin
    num_aux:=vector_numeros[pos_max];
    vector_numeros[pos_max]:=vector_numeros[pos_min];
    vector_numeros[pos_min]:=num_aux;
end;

```

```
var
  vector_numeros: t_vector_numeros;
  numeros, ele_max, pos_max, ele_min, pos_min: int16;
begin
  randomize;
  numeros:=0;
  ele_max:=low(int16); pos_max:=0;
  ele_min:=high(int16); pos_min:=0;
  cargar_vector_numeros(vector_numeros,numeros);
  if (numeros>0) then
    begin
      elementoMaximo(vector_numeros,numeros,ele_max,pos_max);
      elementoMinimo(vector_numeros,numeros,ele_min,pos_min);
      intercambio(vector_numeros,pos_max,pos_min);
      textcolor(green); write('El elemento máximo '); textcolor(red); write(ele_max);
textcolor(green); write(', que se encontraba en la posición '); textcolor(red);
write(pos_max); textcolor(green); write(', fue intercambiado con el elemento mínimo ');
textcolor(red); write(ele_min); textcolor(green); write(', que se encontraba en la posición
'); textcolor(red); write(pos_min);
    end;
  end.
```

## Ejercicio 6.

*Dado que en la solución anterior se recorre dos veces el vector (una para calcular el elemento máximo y otra para el mínimo), implementar un único módulo que recorra una única vez el vector y devuelva ambas posiciones.*

```

program TP4_E6;
{$codepage UTF8}
uses crt;
const
  num_total=100;
  num_salida=0;
type
  t_numero=1..num_total;
  t_vector_numeros=array[t_numero] of int16;
procedure cargar_vector_numeros(var vector_numeros: t_vector_numeros; var numeros: int16);
var
  num: int16;
begin
  num:=num_salida+random(101);
  while ((num<>num_salida) and (numeros<num_total)) do
  begin
    numeros:=numeros+1;
    vector_numeros[numeros]:=num;
    num:=num_salida+random(101);
  end;
end;
procedure elementosMaximoYMinimo(vector_numeros: t_vector_numeros; numeros: int16; var
ele_max, pos_max, ele_min, pos_min: int16);
var
  i: t_numero;
begin
  for i:= 1 to numeros do
  begin
    if (vector_numeros[i]>ele_max) then
    begin
      ele_max:=vector_numeros[i];
      pos_max:=i;
    end;
    if (vector_numeros[i]<ele_min) then
    begin
      ele_min:=vector_numeros[i];
      pos_min:=i;
    end;
  end;
end;
procedure intercambio(var vector_numeros: t_vector_numeros; pos_max, pos_min: int16);
var
  num_aux: int16;
begin
  num_aux:=vector_numeros[pos_max];
  vector_numeros[pos_max]:=vector_numeros[pos_min];
  vector_numeros[pos_min]:=num_aux;
end;
var
  vector_numeros: t_vector_numeros;
  numeros, ele_max, pos_max, ele_min, pos_min: int16;
begin
  randomize;
  numeros:=0;
  ele_max:=low(int16); pos_max:=0;
  ele_min:=high(int16); pos_min:=0;
  cargar_vector_numeros(vector_numeros,numeros);

```

```
if (numeros>0) then
begin
  elementosMaximoYMinimo(vector_numeros,numeros,ele_max,pos_max,ele_min,pos_min);
  intercambio(vector_numeros,pos_max,pos_min);
  textcolor(green); write('El elemento máximo '); textcolor(red); write(ele_max);
textcolor(green); write(', que se encontraba en la posición '); textcolor(red);
write(pos_max); textcolor(green); write(', fue intercambiado con el elemento mínimo ');
textcolor(red); write(ele_min); textcolor(green); write(', que se encontraba en la posición
'); textcolor(red); write(pos_min);
  end;
end.
```

## Ejercicio 7.

Realizar un programa que lea números enteros desde teclado hasta que se ingrese el valor -1 (que no debe procesarse) e informe:

- la cantidad de ocurrencias de cada dígito procesado.
- el dígito más leído.
- los dígitos que no tuvieron ocurrencias.

Por ejemplo, si la secuencia que se lee es: 63 34 99 94 96 -1, el programa deberá informar:

- Número 3: 2 veces
- Número 4: 2 veces
- Número 6: 2 veces
- Número 9: 4 veces
- El dígito más leído fue el 9
- Los dígitos que no tuvieron ocurrencias son: 0, 1, 2, 5, 7, 8

```

program TP4_E7;
{$codepage UTF8}
uses crt;
const
  digitos_total=9;
  num_salida=-1;
  num_corte=0;
type
  t_digito=0..digitos_total;
  t_vector_cantidades=array[t_digito] of int16;
procedure inicializar_vector_cantidades(var vector_cantidades: t_vector_cantidades);
var
  i: t_digito;
begin
  for i:= 1 to digitos_total do
    vector_cantidades[i]:=0;
  end;
procedure descomponer_numero(var vector_cantidades: t_vector_cantidades; num: int16);
var
  digito: t_digito;
begin
  if (num=0) then
    vector_cantidades[0]:=vector_cantidades[0]+1
  else
    while (num<>0) do
      begin
        digito:=num mod 10;
        vector_cantidades[digito]:=vector_cantidades[digito]+1;
        num:=num div 10;
      end;
  end;
procedure cargar_vector_cantidades(var vector_cantidades: t_vector_cantidades);
var
  num: int8;
begin
  num:=num_salida+random(high(int8));
  while (num<>num_salida) do
    begin
      descomponer_numero(vector_cantidades,num);
    end;
  end;

```

```

    num:=num_salida+random(102);
  end;
end;
procedure digito_mas_leido(num: int16; digito: int8; var num_max: int16; var digito_max:
int8);
begin
  if (num>num_max) then
  begin
    num_max:=num;
    digito_max:=digito;
  end;
end;
procedure digitos_sin_ocurrencias(num: int16; digito: int8; var vector_cantidades2:
t_vector_cantidades; var dimL: int8);
begin
  if (num=num_corte) then
  begin
    dimL:=dimL+1;
    vector_cantidades2[dimL]:=digito;
  end;
end;
procedure procesar_vector_cantidades(vector_cantidades: t_vector_cantidades);
var
  i: t_digito;
  digito_max, dimL: int8;
  num_max: int16;
  vector_cantidades2: t_vector_cantidades;
begin
  num_max:=low(int16); digito_max:=-1;
  dimL:=0;
  for i:= 0 to digitos_total do
  begin
    textcolor(green); write('Número ',i,' : '); textcolor(red); write(vector_cantidades[i]);
textcolor(green); writeln(' veces');
    digito_mas_leido(vector_cantidades[i],i,num_max,digito_max);
    digitos_sin_ocurrencias(vector_cantidades[i],i,vector_cantidades2,dimL);
  end;
  textcolor(green); write('El dígito más leído fue el '); textcolor(red); writeln(digito_max);
  if (dimL>0) then
  begin
    textcolor(green); write('Los dígitos que no tuvieron ocurrencias son: ');
    for i:= 1 to dimL do
    begin
      if (i<dimL) then
      begin
        textcolor(red); write(vector_cantidades2[i]); textcolor(green); write(', ');
      end
      else
      begin
        textcolor(red); write(vector_cantidades2[i]);
      end;
    end;
  end;
  else
  begin
    textcolor(red); write('No hay dígitos sin ocurrencias');
  end;
end;
var
  vector_cantidades: t_vector_cantidades;
begin
  randomize;
  inicializar_vector_cantidades(vector_cantidades);
  cargar_vector_cantidades(vector_cantidades);
  procesar_vector_cantidades(vector_cantidades);
end.

```



## Ejercicio 8.

Realizar un programa que lea y almacene la información de 400 alumnos ingresantes a la Facultad de Informática de la UNLP en el año 2020. De cada alumno, se lee: número de inscripción, DNI, apellido, nombre y año de nacimiento. Una vez leída y almacenada toda la información, calcular e informar:

- El porcentaje de alumnos con DNI compuesto sólo por dígitos pares.
- Apellido y nombre de los dos alumnos de mayor edad.

```
program TP4_E8;
{$codepage UTF8}
uses crt;
const
  alumnos_total=400;
type
  t_alumno=1..alumnos_total;
  t_registro_alumno=record
    numero: int16;
    dni: int32;
    apellido: string;
    nombre: string;
    nacimiento: int16;
  end;
  t_vector_alumnos=array[t_alumno] of t_registro_alumno;
function random_string(length: int8): string;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
    random_string:=string_aux;
  end;
end;
procedure leer_alumno(var registro_alumno: t_registro_alumno);
begin
  registro_alumno.numero:=1+random(high(int16));
  registro_alumno.dni:=10000000+random(40000001);
  registro_alumno.apellido:=random_string(5+random(6));
  registro_alumno.nombre:=random_string(5+random(6));
  registro_alumno.nacimiento:=(2020-18)-random(10);
end;
procedure cargar_vector_alumnos(var vector_alumnos: t_vector_alumnos);
var
  registro_alumno: t_registro_alumno;
  i: t_alumno;
begin
  for i:= 1 to alumnos_total do
    begin
      leer_alumno(registro_alumno);
      vector_alumnos[i]:=registro_alumno;
    end;
  end;
end;
function hay_impar(dni: int32): boolean;
var
  digito: int8;
  impar: boolean;
begin
  impar:=false;
  while ((dni<>0) and (impar<>true)) do
```

```

begin
    digito:=dni mod 10;
    impar:=(digito mod 2<>0);
    dni:=dni div 10;
end;
hay_impar:=impar;
end;
procedure actualizar_minimos(nacimiento: int16; apellido, nombre: string; var
nacimiento_min1, nacimiento_min2: int16; var apellido_min1, nombre_min1, apellido_min2,
nombre_min2: string);
begin
    if (nacimiento<nacimiento_min1) then
    begin
        nacimiento_min2:=nacimiento_min1;
        apellido_min2:=apellido_min1;
        nombre_min2:=nombre_min1;
        nacimiento_min1:=nacimiento;
        apellido_min1:=apellido;
        nombre_min1:=nombre;
    end
    else
        if (nacimiento<nacimiento_min2) then
        begin
            nacimiento_min2:=nacimiento;
            apellido_min2:=apellido;
            nombre_min2:=nombre;
        end;
    end;
end;
procedure procesar_vector_alumnos(vector_alumnos: t_vector_alumnos; var porcentaje_pares:
real; var apellido_min1, nombre_min1, apellido_min2, nombre_min2: string);
var
    i: t_alumno;
    alumnos_pares, nacimiento_min1, nacimiento_min2: int16;
begin
    alumnos_pares:=0;
    nacimiento_min1:=high(int16); nacimiento_min2:=high(int16);
    for i:= 1 to alumnos_total do
    begin
        if (hay_impar(vector_alumnos[i].dni)=false) then
            alumnos_pares:=alumnos_pares+1;
        actualizar_minimos(vector_alumnos[i].nacimiento,vector_alumnos[i].apellido,vector_alumnos[
i].nombre,nacimiento_min1,nacimiento_min2,apellido_min1,nombre_min1,apellido_min2,nombre_min2)
        ;
    end;
    porcentaje_pares:=alumnos_pares/alumnos_total*100;
end;
var
    vector_alumnos: t_vector_alumnos;
    porcentaje_pares: real;
    apellido_min1, nombre_min1, apellido_min2, nombre_min2: string;
begin
    randomize;
    porcentaje_pares:=0;
    apellido_min1:=''; nombre_min1:=''; apellido_min2:=''; nombre_min2:='';
    cargar_vector_alumnos(vector_alumnos);
    procesar_vector_alumnos(vector_alumnos,porcentaje_pares,apellido_min1,nombre_min1,apellido_m
in2,nombre_min2);
    textcolor(green); write('El porcentaje de alumnos con DNI compuesto sólo por dígitos pares
es '); textcolor(red); write(porcentaje_pares:0:2); textcolor(green); writeln('%');
    textcolor(green); write('El apellido y nombre de los dos alumnos de mayor edad son ');
    textcolor(red); write(apellido_min1,' ',nombre_min1); textcolor(green); write(' y ');
    textcolor(red); write(apellido_min2,' ',nombre_min2);
end.

```

## Ejercicio 9.

Modificar la solución del punto anterior considerando que el programa lea y almacene la información de, a lo sumo, 400 alumnos. La lectura finaliza cuando se ingresa el DNI -1 (que no debe procesarse).

```

program TP4_E9;
{$codepage UTF8}
uses crt;
const
  alumnos_total=400;
  dni_salida=-1;
type
  t_alumno=1..alumnos_total;
  t_registro_alumno=record
    numero: int16;
    dni: int32;
    apellido: string;
    nombre: string;
    nacimiento: int16;
  end;
  t_vector_alumnos=array[t_alumno] of t_registro_alumno;
function random_string(length: int8): string;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_alumno(var registro_alumno: t_registro_alumno);
var
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    registro_alumno.dni:=dni_salida
  else
    registro_alumno.dni:=10000000+random(40000001);
    if (registro_alumno.dni<>dni_salida) then
      begin
        registro_alumno.numero:=1+random(high(int16));
        registro_alumno.apellido:=random_string(5+random(6));
        registro_alumno.nombre:=random_string(5+random(6));
        registro_alumno.nacimiento:=(2020-18)-random(10);
      end;
  end;
end;
procedure cargar_vector_alumnos(var vector_alumnos: t_vector_alumnos; var alumnos: int16);
var
  registro_alumno: t_registro_alumno;
begin
  leer_alumno(registro_alumno);
  while ((registro_alumno.dni<>dni_salida) and (alumnos<alumnos_total)) do
    begin
      alumnos:=alumnos+1;
      vector_alumnos[alumnos]:=registro_alumno;
      leer_alumno(registro_alumno);
    end;
  end;
end;
function hay_impar(dni: int32): boolean;
var

```

```

    digito: int8;
    impar: boolean;
begin
    impar:=false;
    while ((dni<>0) and (impar<>true)) do
    begin
        digito:=dni mod 10;
        impar:=(digito mod 2<>0);
        dni:=dni div 10;
    end;
    hay_impar:=impar;
end;

procedure actualizar_minimos(nacimiento: int16; apellido, nombre: string; var
nacimiento_min1, nacimiento_min2: int16; var apellido_min1, nombre_min1, apellido_min2,
nombre_min2: string);
begin
    if (nacimiento<nacimiento_min1) then
    begin
        nacimiento_min2:=nacimiento_min1;
        apellido_min2:=apellido_min1;
        nombre_min2:=nombre_min1;
        nacimiento_min1:=nacimiento;
        apellido_min1:=apellido;
        nombre_min1:=nombre;
    end
    else
        if (nacimiento<nacimiento_min2) then
        begin
            nacimiento_min2:=nacimiento;
            apellido_min2:=apellido;
            nombre_min2:=nombre;
        end;
    end;

procedure procesar_vector_alumnos(vector_alumnos: t_vector_alumnos; alumnos: int16; var
porcentaje_pares: real; var apellido_min1, nombre_min1, apellido_min2, nombre_min2: string);
var
    i: t_alumno;
    alumnos_pares, nacimiento_min1, nacimiento_min2: int16;
begin
    alumnos_pares:=0;
    nacimiento_min1:=high(int16); nacimiento_min2:=high(int16);
    for i:= 1 to alumnos do
    begin
        if (hay_impar(vector_alumnos[i].dni)=false) then
            alumnos_pares:=alumnos_pares+1;
        actualizar_minimos(vector_alumnos[i].nacimiento,vector_alumnos[i].apellido,vector_alumnos[
i].nombre,nacimiento_min1,nacimiento_min2,apellido_min1,nombre_min1,apellido_min2,nombre_min2)
        ;
    end;
    if (alumnos>0) then
        porcentaje_pares:=alumnos_pares/alumnos*100;
    end;
var
    vector_alumnos: t_vector_alumnos;
    alumnos: int16;
    porcentaje_pares: real;
    apellido_min1, nombre_min1, apellido_min2, nombre_min2: string;
begin
    randomize;
    alumnos:=0;
    porcentaje_pares:=0;
    apellido_min1:=''; nombre_min1:=''; apellido_min2:=''; nombre_min2:='';
    cargar_vector_alumnos(vector_alumnos,alumnos);
    if (alumnos>0) then
    begin

```

```
    procesar_vector_alumnos(vector_alumnos,alumnos,porcentaje_pares,apellido_min1,nombre_min1,
apellido_min2,nombre_min2);
    textcolor(green); write('El porcentaje de alumnos con DNI compuesto sólo por dígitos pares
es '); textcolor(red); write(porcentaje_pares:0:2); textcolor(green); writeln('%');
    textcolor(green); write('El apellido y nombre de los dos alumnos de mayor edad son ');
textcolor(red); write(apellido_min1,' ',nombre_min1); textcolor(green); write(' y ');
textcolor(red); write(apellido_min2,' ',nombre_min2);
    end;
end.
```

**Ejercicio 10.**

Realizar un programa que lea y almacene el salario de los empleados de una empresa de turismo (a lo sumo, 300 empleados). La carga finaliza cuando se lea el salario 0 (que no debe procesarse) o cuando se completa el vector. Una vez finalizada la carga de datos, se pide:

(a) Realizar un módulo que incremente el salario de cada empleado en un 15%. Para ello, implementar un módulo que reciba como parámetro un valor real  $X$ , el vector de valores reales y su dimensión lógica y retorne el mismo vector en el cual cada elemento fue multiplicado por el valor  $X$ .

(b) Realizar un módulo que muestre en pantalla el sueldo promedio de los empleados de la empresa.

```
program TP4_E10;
{$codepage UTF8}
uses crt;
const
    empleados_total=300;
    salario_salida=0;
    incremento=1.15;
type
    t_empleado=1..empleados_total;
    t_vector_salarios=array[t_empleado] of real;
procedure inicializar_vector_salarios(var vector_salarios: t_vector_salarios);
var
    i: t_empleado;
begin
    for i:= 1 to empleados_total do
        vector_salarios[i]:=0;
    end;
procedure cargar_vector_salarios(var vector_salarios: t_vector_salarios; var empleados:
int16);
var
    salario: real;
begin
    salario:=salario_salida+random(1001)/10;
    while ((salario<>salario_salida) and (empleados<empleados_total)) do
        begin
            empleados:=empleados+1;
            vector_salarios[empleados]:=salario;
            salario:=salario_salida+random(1001)/10;
        end;
    end;
procedure incrementar_salarios(incremento: real; var vector_salarios: t_vector_salarios;
empleados: int16);
var
    i: t_empleado;
begin
    for i:= 1 to empleados do
        vector_salarios[i]:=vector_salarios[i]*incremento;
    end;
procedure calcular_salario_promedio(vector_salarios: t_vector_salarios; empleados: int16);
var
    i: t_empleado;
    salario_total, salario_prom: real;
begin
    salario_total:=0;
    for i:= 1 to empleados do
```

```
    salario_total:=salario_total+vector_salarios[i];
    salario_prom:=salario_total/empleados;
    textcolor(green); write('El sueldo promedio de los empleados de la empresa es $');
textcolor(red); write(salario_prom:0:2);
end;
var
    vector_salarios: t_vector_salarios;
    empleados: int16;
begin
    randomize;
    empleados:=0;
    inicializar_vector_salarios(vector_salarios);
    cargar_vector_salarios(vector_salarios,empleados);
    if (empleados>0) then
    begin
        incrementar_salarios(incremento,vector_salarios,empleados);
        calcular_salario_promedio(vector_salarios,empleados);
    end;
end.
```

## Ejercicio 11.

El colectivo de fotógrafos ArgenPics desea conocer los gustos de sus seguidores en las redes sociales. Para ello, para cada una de las 200 fotos publicadas en su página de Facebook, cuenta con la siguiente información: título de la foto, el autor de la foto, cantidad de Me gusta, cantidad de clics y cantidad de comentarios de usuarios. Realizar un programa que lea y almacene esta información. Una vez finalizada la lectura, el programa debe procesar los datos e informar:

- Título de la foto más vista (la que posee mayor cantidad de clics).
- Cantidad total de Me gusta recibidos a las fotos cuyo autor es el fotógrafo “Art Vandelay”.
- Cantidad de comentarios recibidos para cada una de las fotos publicadas en esa página.

```

program TP4_E11;
{$codepage UTF8}
uses crt;
const
  fotos_total=200;
  autor_corte='Art Vandelay';
type
  t_foto=1..fotos_total;
  t_registro_foto=record
    titulo: string;
    autor: string;
    megusta: int16;
    clics: int16;
    comentarios: int16;
  end;
  t_vector_fotos=array[t_foto] of t_registro_foto;
function random_string(length: int8): string;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_foto(var registro_foto: t_registro_foto);
var
  i: int8;
begin
  registro_foto.titulo:=random_string(5+random(6));
  i:=random(10);
  if (i=0) then
    registro_foto.autor:=autor_corte
  else
    registro_foto.autor:=random_string(5+random(6));
  registro_foto.megusta:=random(10001);
  registro_foto.clics:=random(10001);
  registro_foto.comentarios:=random(10001);
end;
procedure cargar_vector_fotos(var vector_fotos: t_vector_fotos);
var
  registro_foto: t_registro_foto;
  i: t_foto;
begin

```



```

    for i:= 1 to fotos_total do
    begin
        leer_foto(registro_foto);
        vector_fotos[i]:=registro_foto;
    end;
end;
procedure actualizar_maximo(clics: int16; titulo: string; var clics_max: int16; var
titulo_max: string);
begin
    if (clics>clics_max) then
    begin
        clics_max:=clics;
        titulo_max:=titulo;
    end;
end;
procedure procesar_vector_fotos(vector_fotos: t_vector_fotos; var titulo_max: string; var
megusta_corte: int16);
var
    i: t_foto;
    clics_max: int16;
begin
    clics_max:=low(int16);
    for i:= 1 to fotos_total do
    begin
        actualizar_maximo(vector_fotos[i].clics,vector_fotos[i].titulo,clics_max,titulo_max);
        if (vector_fotos[i].autor=autor_corte) then
            megusta_corte:=megusta_corte+1;
        textcolor(green); write('La cantidad de comentarios recibidos de la foto ');
textcolor(red); write(vector_fotos[i].titulo); textcolor(green); write(' es ');
textcolor(red); writeln(vector_fotos[i].comentarios);
    end;
end;
var
    vector_fotos: t_vector_fotos;
    megusta_corte: int16;
    titulo_max: string;
begin
    randomize;
    titulo_max:='';
    megusta_corte:=0;
    cargar_vector_fotos(vector_fotos);
    procesar_vector_fotos(vector_fotos,titulo_max,megusta_corte);
    textcolor(green); write('El título de la foto más vista (la que posee mayor cantidad de
clics) es '); textcolor(red); writeln(titulo_max);
    textcolor(green); write('La cantidad total de Me gusta recibidas a las fotos cuyo autor es
el fotógrafo '); textcolor(yellow); write(autor_corte); textcolor(green); write('" es ');
textcolor(red); write(megusta_corte);
end.

```

## Ejercicio 12.

En astrofísica, una galaxia se identifica por su nombre, su tipo (1. elíptica; 2. espiral; 3. lenticular; 4. irregular), su masa (medida en kg) y la distancia en pársecs (pc) medida desde la Tierra. La Unión Astronómica Internacional cuenta con datos correspondientes a las 53 galaxias que componen el Grupo Local. Realizar un programa que lea y almacene estos datos y, una vez finalizada la carga, informe:

- La cantidad de galaxias de cada tipo.
- La masa total acumulada de las 3 galaxias principales (la Vía Láctea, Andrómeda y Triángulo) y el porcentaje que esto representa respecto a la masa de todas las galaxias.
- La cantidad de galaxias no irregulares que se encuentran a menos de 1000 pc.
- El nombre de las dos galaxias con mayor masa y el de las dos galaxias con menor masa.

```
program TP4_E12;
{$codepage UTF8}
uses crt;
const
    galaxias_total=53;
    tipo_ini=1; tipo_fin=4;
    galaxia_corte1='la Via Lactea'; galaxia_corte2='Andromeda'; galaxia_corte3='Triangulo';
    tipo_corte=4; distancia_corte=1000.0;
type
    t_galaxia=1..galaxias_total;
    t_tipo=tipo_ini..tipo_fin;
    t_registro_galaxia=record
        nombre: string;
        tipo: t_tipo;
        masa: real;
        distancia: real;
    end;
    t_vector_galaxias=array[t_galaxia] of t_registro_galaxia;
    t_vector_cantidades=array[t_tipo] of int16;
procedure inicializar_vector_cantidades(var vector_cantidades: t_vector_cantidades);
var
    i: t_tipo;
begin
    for i:= tipo_ini to tipo_fin do
        vector_cantidades[i]:=0;
    end;
function random_string(length: int8): string;
var
    i: int8;
    string_aux: string;
begin
    string_aux:='';
    for i:= 1 to length do
        string_aux:=string_aux+chr(ord('A')+random(26));
        random_string:=string_aux;
    end;
procedure leer_galaxia(var registro_galaxia: t_registro_galaxia);
var
    i: int8;
begin
    i:=random(4);
    if (i=0) then
        registro_galaxia.nombre:=galaxia_corte1
```

```

else if (i=1) then
    registro_galaxia.nombre:=galaxia_corte2
else if (i=2) then
    registro_galaxia.nombre:=galaxia_corte3
else
    registro_galaxia.nombre:='Galaxia '+random_string(5+random(6));
    registro_galaxia.tipo:=tipo_ini+random(tipo_fin);
    registro_galaxia.masa:=1+random(99991)/10;
    registro_galaxia.masa:=1+random(99991)/10;
end;
procedure cargar_vector_galaxias(var vector_galaxias: t_vector_galaxias);
var
    registro_galaxia: t_registro_galaxia;
    i: t_galaxia;
begin
    for i:= 1 to galaxias_total do
        begin
            leer_galaxia(registro_galaxia);
            vector_galaxias[i]:=registro_galaxia;
        end;
    end;
end;
procedure actualizar_maximos(masa: real; nombre: string; var masa_max1, masa_max2: real; var
nombre_max1, nombre_max2: string);
begin
    if (masa>masa_max1) then
        begin
            masa_max2:=masa_max1;
            nombre_max2:=nombre_max1;
            masa_max1:=masa;
            nombre_max1:=nombre;
        end
    else
        if (masa>masa_max2) then
            begin
                masa_max2:=masa;
                nombre_max2:=nombre;
            end;
        end;
    end;
end;
procedure actualizar_minimos(masa: real; nombre: string; var masa_min1, masa_min2: real; var
nombre_min1, nombre_min2: string);
begin
    if (masa<masa_min1) then
        begin
            masa_min2:=masa_min1;
            nombre_min2:=nombre_min1;
            masa_min1:=masa;
            nombre_min1:=nombre;
        end
    else
        if (masa<masa_min2) then
            begin
                masa_min2:=masa;
                nombre_min2:=nombre;
            end;
        end;
    end;
end;
procedure procesar_vector_galaxias(vector_galaxias: t_vector_galaxias; var vector_cantidades:
t_vector_cantidades; var masa_principales, porcentaje_principales: real; var galaxias_corte:
int8; var nombre_max1, nombre_max2, nombre_min1, nombre_min2: string);
var
    i: t_galaxia;
    masa_total, masa_max1, masa_max2, masa_min1, masa_min2: real;
begin
    masa_total:=0;
    masa_max1:=-9999999; masa_max2:=-9999999;
    masa_min1:=9999999; masa_min2:=9999999;
    for i:= 1 to galaxias_total do

```

```

begin
    vector_cantidades[vector_galaxias[i].tipo]:=vector_cantidades[vector_galaxias[i].tipo]+1;
    masa_total:=masa_total+vector_galaxias[i].masa;
    if ((vector_galaxias[i].nombre=galaxia_corte1) or
(vector_galaxias[i].nombre=galaxia_corte2) or (vector_galaxias[i].nombre=galaxia_corte3)) then
        masa_principales:=masa_principales+vector_galaxias[i].masa;
        if ((vector_galaxias[i].tipo<>tipo_corte) and
(vector_galaxias[i].distancia<distancia_corte)) then
            galaxias_corte:=galaxias_corte+1;
            actualizar_maximos(vector_galaxias[i].masa,vector_galaxias[i].nombre,masa_max1,masa_max2,n
ombre_max1,nombre_max2);
            actualizar_minimos(vector_galaxias[i].masa,vector_galaxias[i].nombre,masa_min1,masa_min2,n
ombre_min1,nombre_min2);
        end;
        porcentaje_principales:=masa_principales/masa_total*100;
end;
var
    vector_cantidades_string: array[t_tipo] of string=('eliptica', 'espiral', 'lenticular',
'irregular');
    vector_galaxias: t_vector_galaxias;
    vector_cantidades: t_vector_cantidades;
    galaxias_corte: int8;
    masa_principales, porcentaje_principales: real;
    nombre_max1, nombre_max2, nombre_min1, nombre_min2: string;
begin
    randomize;
    inicializar_vector_cantidades(vector_cantidades);
    masa_principales:=0; porcentaje_principales:=0;
    galaxias_corte:=0;
    nombre_max1:=''; nombre_max2:='';
    nombre_min1:=''; nombre_min2:='';
    cargar_vector_galaxias(vector_galaxias);
    procesar_vector_galaxias(vector_galaxias,vector_cantidades,masa_principales,porcentaje_princ
ipales,galaxias_corte,nombre_max1,nombre_max2,nombre_min1,nombre_min2);
    textcolor(green); write('La cantidad de galaxias de cada tipo (1. elíptica; 2. espiral; 3.
lenticular; 4. irregular) es '); textcolor(red); write(vector_cantidades[1],',
',vector_cantidades[2],',',',vector_cantidades[3],',',',vector_cantidades[4]); textcolor(green);
writeln(', respectivamente');
    textcolor(green); write('La masa total acumulada de las 3 galaxias principales (');
textcolor(yellow); write(galaxia_corte1,', ',galaxia_corte2,', ',galaxia_corte3);
textcolor(green); write(') y el porcentaje que esto representa respecto a la masa de todas las
galaxias son '); textcolor(red); write(masa_principales:0:2); textcolor(green); write(' y ');
textcolor(red); write(porcentaje_principales:0:2); textcolor(green); writeln('%
respectivamente');
    textcolor(green); write('La cantidad de galaxias no '); textcolor(yellow);
write(vector_cantidades_string[tipo_corte]); textcolor(green); write(' que se encuentran a
menos de '); textcolor(yellow); write(distancia_corte:0:2); textcolor(green); write(' pc es
'); textcolor(red); writeln(galaxias_corte);
    textcolor(green); write('Los nombres de las dos galaxias con mayor masa son ');
textcolor(red); write(nombre_max1); textcolor(green); write(' y '); textcolor(red);
writeln(nombre_max2);
    textcolor(green); write('Los nombres de las dos galaxias con menor masa son ');
textcolor(red); write(nombre_min1); textcolor(green); write(' y '); textcolor(red);
write(nombre_min2);
end.

```

### Ejercicio 13.

*El Grupo Intergubernamental de Expertos sobre el Cambio Climático de la ONU (IPCC) realiza todos los años mediciones de temperatura en 100 puntos diferentes del planeta y, para cada uno de estos lugares, obtiene un promedio anual. Este relevamiento se viene realizando desde hace 50 años y, con todos los datos recolectados, el IPCC se encuentra en condiciones de realizar análisis estadísticos. Realizar un programa que lea y almacene los datos correspondientes a las mediciones de los últimos 50 años (la información se ingresa ordenada por año). Una vez finalizada la carga de la información, obtener:*

- *El año con mayor temperatura promedio a nivel mundial.*
- *El año con la mayor temperatura detectada en algún punto del planeta en los últimos 50 años.*

```

program TP4_E13;
{$codepage UTF8}
uses crt;
const
  puntos_total=100;
  anio_ini=1974; anio_fin=2023;
type
  t_punto=1..puntos_total;
  t_anio=anio_ini..anio_fin;
  t_vector_puntos=array[t_punto] of real;
  t_vector_anios=array[t_anio] of t_vector_puntos;
procedure inicializar_vector_anios(var vector_anios: t_vector_anios);
var
  i: t_anio;
  j: t_punto;
begin
  for i:= anio_ini to anio_fin do
    for j:= 1 to puntos_total do
      vector_anios[i][j]:=0;
    end;
  end;
procedure cargar_vector_anios(var vector_anios: t_vector_anios);
var
  i: t_anio;
  j: t_punto;
begin
  for i:= anio_ini to anio_fin do
    for j:= 1 to puntos_total do
      vector_anios[i][j]:=-50+random(1001)/10;
    end;
  end;
procedure actualizar_maximo1(promedio_anio: real; anio: int16; var promedio_max: real; var anio_max1: int16);
begin
  if (promedio_anio>promedio_max) then
    begin
      promedio_max:=promedio_anio;
      anio_max1:=anio;
    end;
  end;
procedure actualizar_maximo2(temperatura: real; anio: int16; var temperatura_max: real; var anio_max2: int16);
begin
  if (temperatura>temperatura_max) then
    begin
      temperatura_max:=temperatura;
      anio_max2:=anio;
    end;
  end;

```

```
    end;
end;
procedure procesar_vector_anios(vector_anios: t_vector_anios; var anio_max1, anio_max2:
int16);
var
    i: t_anio;
    j: t_punto;
    temperatura_anio, promedio_anio, promedio_max, temperatura_max: real;
begin
    promedio_max:=-9999999;
    temperatura_max:=-9999999;
    for i:= anio_ini to anio_fin do
        begin
            temperatura_anio:=0;
            for j:= 1 to puntos_total do
                begin
                    temperatura_anio:=temperatura_anio+vector_anios[i][j];
                    actualizar_maximo2(vector_anios[i][j],i,temperatura_max,anio_max2);
                end;
            promedio_anio:=temperatura_anio/puntos_total;
            actualizar_maximo1(promedio_anio,i,promedio_max,anio_max1);
        end;
    end;
end;
var
    vector_anios: t_vector_anios;
    anio_max1, anio_max2: int16;
begin
    randomize;
    anio_max1:=0; anio_max2:=0;
    inicializar_vector_anios(vector_anios);
    cargar_vector_anios(vector_anios);
    procesar_vector_anios(vector_anios,anio_max1,anio_max2);
    textcolor(green); write('El año con mayor temperatura promedio a nivel mundial es ');
textcolor(red); writeln(anio_max1);
    textcolor(green); write('El año con la mayor temperatura detectada en algún punto del
planeta en los últimos 50 años es '); textcolor(red); write(anio_max2);
end.
```

**Ejercicio 14.**

El repositorio de código fuente más grande en la actualidad, GitHub, desea estimar el monto invertido en los proyectos que aloja. Para ello, dispone de una tabla con información de los desarrolladores que participan en un proyecto de software, junto al valor promedio que se paga por hora de trabajo:

CÓDIGO	ROL DEL DESARROLLADOR	VALOR/HORA (USD)
1	Analista Funcional	35,20
2	Programador	27,45
3	Administrador de bases de datos	31,03
4	Arquitecto de software	44,28
5	Administrador de redes y seguridad	39,87

*Nota: los valores/hora se incluyen a modo de ejemplo*

Realizar un programa que procese la información de los desarrolladores que participaron en los 1000 proyectos de software más activos durante el año 2017. De cada participante, se conoce su país de residencia, código de proyecto (1 a 1000), el nombre del proyecto en el que participó, el rol que cumplió en dicho proyecto (1 a 5) y la cantidad de horas trabajadas. La lectura finaliza al ingresar el código de proyecto -1, que no debe procesarse. Al finalizar la lectura, el programa debe informar:

- El monto total invertido en desarrolladores con residencia en Argentina.
- La cantidad total de horas trabajadas por Administradores de bases de datos.
- El código del proyecto con menor monto invertido.
- La cantidad de Arquitectos de software de cada proyecto.

```
program TP4_E14;
{$codepage UTF8}
uses crt;
const
  proyecto_ini=1; proyecto_fin=1000;
  rol_ini=1; rol_fin=5;
  proyecto_salida=-1;
  pais_corte='Argentina';
  rol_corte1=3;
  rol_corte2=4;
type
  t_proyecto=proyecto_ini..proyecto_fin;
  t_rol=rol_ini..rol_fin;
  t_registro_participante=record
    pais: string;
    proyecto: int16;
    nombre: string;
    rol: t_rol;
    horas: int16;
  end;
```

```

t_registro_proyecto=record
    monto: real;
    cantidad: int16;
end;
t_vector_proyectos=array[t_proyecto] of t_registro_proyecto;
t_vector_salarios=array[t_rol] of real;
procedure cargar_vector_salarios(var vector_salarios: t_vector_salarios);
begin
    vector_salarios[1]:=35.20;
    vector_salarios[2]:=27.45;
    vector_salarios[3]:=31.03;
    vector_salarios[4]:=44.28;
    vector_salarios[5]:=39.87;
end;
procedure inicializar_vector_proyectos(var vector_proyectos: t_vector_proyectos);
var
    i: t_proyecto;
begin
    for i:= proyecto_ini to proyecto_fin do
        begin
            vector_proyectos[i].monto:=0;
            vector_proyectos[i].cantidad:=0;
        end;
    end;
end;
function random_string(length: int8): string;
var
    i: int8;
    string_aux: string;
begin
    string_aux:='';
    for i:= 1 to length do
        string_aux:=string_aux+chr(ord('A')+random(26));
    end;
    random_string:=string_aux;
end;
procedure leer_participante(var registro_participante: t_registro_participante);
var
    i: int8;
begin
    i:=random(100);
    if (i=0) then
        registro_participante.proyecto:=proyecto_salida
    else
        registro_participante.proyecto:=proyecto_ini+random(proyecto_fin);
    if (registro_participante.proyecto<>proyecto_salida) then
        begin
            i:=random(10);
            if (i=0) then
                registro_participante.pais:=pais_corte
            else
                registro_participante.pais:=random_string(5+random(6));
                registro_participante.nombre:=random_string(5+random(6));
                registro_participante.rol:=rol_ini+random(rol_fin);
                registro_participante.horas:=1+random(100);
            end;
        end;
    end;
end;
procedure cargar_vector_proyectos(var vector_proyectos: t_vector_proyectos; var monto_corte:
real; var horas_corte: int16; vector_salarios: t_vector_salarios);
var
    registro_participante: t_registro_participante;
begin
    leer_participante(registro_participante);
    while (registro_participante.proyecto<>proyecto_salida) do
        begin
            if (registro_participante.pais=pais_corte) then
                monto_corte:=monto_corte+vector_salarios[registro_participante.rol]*registro_participant
e.horas;

```



```

    if (registro_participante.rol=rol_corte1) then
        horas_corte:=horas_corte+registro_participante.horas;
        vector_proyectos[registro_participante.proyecto].monto:=vector_proyectos[registro_participante.proyecto].monto+vector_salarios[registro_participante.rol]*registro_participante.horas;
    if (registro_participante.rol=rol_corte2) then
        vector_proyectos[registro_participante.proyecto].cantidad:=vector_proyectos[registro_participante.proyecto].cantidad+1;
        leer_participante(registro_participante);
    end;
end;
procedure procesar_vector_proyectos(vector_proyectos: t_vector_proyectos; var proyecto_min: int16);
var
    i: t_proyecto;
    monto_min: real;
begin
    monto_min:=9999999;
    for i:= proyecto_ini to proyecto_fin do
        begin
            if ((vector_proyectos[i].monto>0) and (vector_proyectos[i].monto<monto_min)) then
                begin
                    monto_min:=vector_proyectos[i].monto;
                    proyecto_min:=i;
                end;
            if (vector_proyectos[i].cantidad>0) then
                begin
                    textcolor(green); write('La cantidad de Arquitectos de software del proyecto ',i,' es '); textcolor(red); writeln(vector_proyectos[i].cantidad);
                end;
            end;
        end;
end;
var
    vector_salarios: t_vector_salarios;
    vector_proyectos: t_vector_proyectos;
    horas_corte, proyecto_min: int16;
    monto_corte: real;
begin
    randomize;
    cargar_vector_salarios(vector_salarios);
    monto_corte:=0;
    horas_corte:=0;
    proyecto_min:=0;
    inicializar_vector_proyectos(vector_proyectos);
    cargar_vector_proyectos(vector_proyectos,monto_corte,horas_corte,vector_salarios);
    textcolor(green); write('El monto total invertido en desarrolladores con residencia en ');
textcolor(yellow); write(pais_corte); textcolor(green); write(' es U$D '); textcolor(red);
writeln(monto_corte:0:2);
    textcolor(green); write('La cantidad total de horas trabajadas por Administradores de bases de datos es '); textcolor(red); writeln(horas_corte);
    procesar_vector_proyectos(vector_proyectos,proyecto_min);
    textcolor(green); write('El código de proyecto con menor monto invertido es ');
textcolor(red); write(proyecto_min);
end.

```