

Examen - Programación 1

Diciembre 2021

Leer con atención:

- Todos los programas o fragmentos de programas deben ser escritos en el lenguaje Pascal tal como fue dado en el curso.
- En todos los problemas se evaluará, además de la lógica correcta, la utilización de un buen estilo de programación de acuerdo a los criterios impartidos en el curso.
- Entregue solamente las hojas de solución escritas a lápiz.

Ejercicio 1 (30 puntos)

Dada la siguiente declaración de una secuencia de enteros:

```
const
  Max = ..;

type
  IntArr = record
    val : array [1..Max] of integer;
    tope : 0..Max
  end;
```

Definir el procedimiento:

```
procedure zipAdd (a1,a2: IntArr; var a : IntArr);
```

que dadas dos secuencias a_1 y a_2 de tipo `IntArr` retorna otra secuencia a tal que $a[i]$ es igual a $a_1[i] + a_2[i]$. En el caso que a_1 y a_2 sean de distinto largo, por ejemplo a_1 contenga n elementos y a_2 contenga m , con $n > m$, entonces $a[j]$ será igual a $a_1[j]$ para $m + 1 \leq j \leq n$.

Ejemplos: Para `Max = 8` en la siguiente tabla se muestra el arreglo con tope resultante de la invocación de `zipAdd(a1,a2,a)`:

Antes	Después
a1.val: 5 -4 6 ? ? ? ? ? a2.val: 9 0 8 -2 7 ? ? ?	a.val: 14 -4 14 -2 7 ? ? ?
a1.val: 5 -4 6 ? ? ? ? ? a2.val: ? ? ? ? ? ? ? ?	a.val: 5 -4 6 ? ? ? ? ?

Ejercicio 2 (30 puntos)

Se consideran las siguientes declaraciones:

```
const
  Max = ...;
type
  TPar = record
    primero : real;
    segundo : real
  end;
  TArreglo = array [1..Max] of TPar;
```

Se quiere examinar las veces que aparece un cierto valor real como primera componente de los pares en un arreglo dado.

Para representar el resultado de esta búsqueda se define un tipo:

```
type
  TMultiplicidad = (ninguno, uno, dos, muchos);
  TResultado = record case mult : TMultiplicidad of
    ninguno      : ();
    uno           : (v      : real);
    dos           : (v1, v2 : real);
    muchos       : ();
  end;
```

El valor buscado puede no aparecer (como primera componente), aparecer exactamente una vez, exactamente dos veces o más de dos veces ("muchas"). En el caso de que aparezca una vez, el campo *v* es el valor asociado como segunda componente del par. En el caso de que aparezca dos veces, los campos *v1* y *v2* tendrán los correspondientes valores asociados. En el caso de que aparezca 3 o más veces no se guardará la información asociada con el valor.

Se pide un subprograma:

```
procedure BuscarValor (w : real; arreglo : TArreglo; var result : TResultado);
```

que estudie las apariciones de *w* como primera componente de los pares de *arreglo* y retorne *result* con la información tal como se describió más arriba.

Ejemplo: si *Max* = 6 y el arreglo *ta* = [(3.10, 4.50)(4.50, 6.20)(3.10, 8.40)(4.50, 2.30)(54.60, 7.89)(3.10, 4.60)], entonces:

- `BuscarValor(7.89, ta, result)` retorna `result = [mult = ninguno]`
- `BuscarValor(54.60, ta, result)` retorna `result = [mult = uno, v = 7.89]`
- `BuscarValor(4.50, ta, result)` retorna `result = [mult = dos, v1 = 6.20, v2 = 2.30]`
- `BuscarValor(3.10, ta, result)` retorna `result = [mult = muchos]`

Ejercicio 3 (30 puntos)

Considere la siguiente declaración de tipos:

```
type Lista = ^Nodo;  
  Nodo = record  
    elem : integer;  
    sig  : Lista  
  end;
```

Definir un procedimiento que, dada una lista l de tipo `Lista`, retorne la lista resultante de borrar de l la primera celda que contenga un número entero negativo. Si la lista no contiene números negativos el procedimiento retorna la lista incambiada.

```
procedure borrarPrimNegL (var l : Lista);
```

Ejemplo: si la lista

- $l = (-2, 3, -4, 3, 2)$ entonces `borrarPrimNegL(l)` retorna $l = (3, -4, 3, 2)$
- $l = (2, 3, -4, 3, 2)$ entonces `borrarPrimNegL(l)` retorna $l = (2, 3, 3, 2)$
- $l = (2, 3, 4, 3, -2)$ entonces `borrarPrimNegL(l)` retorna $l = (2, 3, 4, 3)$
- $l = (2, 3, 4, 3, 2)$ entonces `borrarPrimNegL(l)` retorna $l = (2, 3, 4, 3, 2)$

Ejercicio 4 (10 puntos)

Indicar cuál es la salida que produce el siguiente programa cuando en la entrada se ingresa el último dígito de su cédula de identidad (nos referimos a la cédula del estudiante). Por ejemplo, si su cédula es 1234567-8 deberá ingresar el número 7.

```
Program ExDic2021;
var x, y, z: integer;

procedure suma (a : integer; var b, c : integer);
  function itera (a, b : integer) : integer;
  var i : integer;
  begin
    for i := 1 to 3 do
      a := a + b;
      itera := a;
    end;
begin
  b := itera (y, b);
  a := x + 8;
  writeln (a, b);
  c := a + 1
end;

function evalua (b: integer): integer;
  procedure pp (bol : boolean; var x : integer);
  begin
    if bol then x:=1 else x :=2;
  end;
  var
    x : Integer;
begin
  pp(b < 5,x);
  evalua := x;
end;

begin
  readln(x);
  y := evalua ( x );
  suma (x, y, z);
  writeln(x, y, z)
end.
```