Examen - Programación 1

Instituto de Computación
Febrero 2024

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.
- Escribir las respuestas de un solo lado de la hoja. Entregar solamente las hojas de solución escritas a lápiz.
- En cada hoja entregada se debe incluir nombre, cédula y qué número de hoja es. En la primera hoja se debe incluir además la cantidad total de hojas entregadas.

Ejercicio 1 (35 puntos)

Se define la siguiente estructura para representar la lista de enteros:

```
type
    LInt = ^Nodo;
    Nodo = record
        val : integer;
        sig : LInt;
    end;
```

a) (15 puntos)

Escribir la función:

```
function producto(lista: LInt) : integer;
```

que computa el producto de los elementos de la lista. El producto de la lista vacía es 1. Tener en cuenta que $0 * n = 0$.

Solución:

```
function producto(lista: LInt) : integer;
var
   res  : integer;
   iter : LInt;
begin
   res  := 1;
   iter := lista;
   while (iter <> NIL) and (res <> 0) do
   begin
      res  := res * iter^.val;
      iter := iter^.sig
   end;
   producto := res
end;
```

b) (20 puntos)

Escribir la función:

```
function rango(ini, fin : integer ) : LInt;
```

que genera una lista ordenada de forma ascendente con los enteros del rango entre 'ini' y 'fin'. Si 'ini' >'fin', entonces se retorna la lista vacía.

Solución:

```
function rango(ini, fin : integer ) : LInt;
var
   res, aux : LInt;
   i   : integer;
begin
   res := NIL;

   for i := fin downto ini do
   begin
      aux := res;
      new(res);
      res^.val := i;
      res^.sig := aux
   end;

   rango := res
end;
```

Ejercicio 2 (49 puntos)

Dadas las siguientes declaraciones:

```
const N = .. ; { N > 0 }
type
   PosibleReal = record case esReal : boolean of
                    true  : ( valor : real);
                    false : ();
                 end;

   Arreglo = array [1 .. N] of PosibleReal;

   ArregloConTope = record
                      info : array [1..N] of real;
                      tope : 0..N;
                    end;
```

a) (14 puntos)

Escribir el procedimiento:

```
procedure ObtenerReales( entrada : Arreglo; var
    salida : ArregloConTope);
```

que toma como `entrada` una estructura del tipo Arreglo sin números repetidos y retorna como `salida` una estructura del tipo ArregloConTope que contiene todos los reales que aparecen en `entrada`.

Solución:

```
procedure ObtenerReales( entrada : Arreglo; var
    salida : ArregloConTope);
var i : integer;
begin
   salida.tope:= 0;
   for i:= 1 to N do
      if entrada[i].esReal then
      begin
         salida.tope:= salida.tope + 1;
         salida.info[salida.tope]:= entrada[i].valor
      end
end;
```

b) (15 puntos)

Escribir la función:

```
function pertenece(elem : real; a : Arreglo) :
    boolean;
```

que verifica si un real `elem` aparece o no en el Arreglo `a`.

Solución:

```
function pertenece(elem : real; a : Arreglo) :
    boolean;
var i : integer;
begin
   i:= 1;
   while (i <= N) and not (a[i].esReal and (a[i].
       valor = elem)) do
      i:= i + 1;

   pertenece:= i <= N
end;
```

c) (20 puntos)

Escribir la función:

```
function cantidadRealesComunes(a,b : Arreglo) :
    integer;
```

que cuenta la cantidad de reales de `a` que también se encuentran en `b`. Asuma que ni `a` ni `b` tienen números reales repetidos. Sugerencia: usar la parte b).

Solución:

```
function cantidadRealesComunes(a,b : Arreglo) :
    integer;
var i, cant : integer;
begin
   cant := 0;

   for i:= 1 to N do
      if a[i].esReal and pertenece(a[i].valor, b)
          then
         cant := cant + 1;

   cantidadRealesComunes := cant
end;
```

Ejercicio 3 (16 puntos)

Determinar la salida del siguiente programa, cuando se le da como entrada el dígito de las unidades (el último dígito antes del guion) de su número de cédula. Por ejemplo, si su cédula es 1234567-8, la entrada será 7.

```
program alcance;

var x, y, z : integer;

procedure uno (x : integer; var y : integer);
var z : integer;
begin
   z := y - x;
   x := 10 - y;
   y := z + 2;
   writeln(x,' ',y,' ',z)
end;

procedure dos (var x : integer; y : integer);
begin
   z := y - x;
   x := 10 - y;
   y := z + 2;
   uno(y,x)
end;

begin
   readln(x);
   y := 10 - x;
   z := x * y;
   uno(z, x);
   dos(x, y);
   writeln(x,' ',z)
end.
```

Solución:

Entrada	Salida		
0	10	2	0
	10	-8	-10
	-8	8	
1	9	-6	-8
	9	-14	-16
	-14	15	
2	8	-12	-14
	8	-18	-20
	-18	20	
3	7	-16	-18
	7	-20	-22
	-20	23	
4	6	-18	-20
	6	-20	-22
	-20	24	

Entrada	Salida		
5	5	-18	-20
	5	-18	-20
	-18	23	
6	4	-16	-18
	4	-14	-16
	-14	20	
7	3	-12	-14
	3	-8	-10
	-8	15	
8	2	-6	-8
	2	0	-2
	0	8	
9	1	2	0
	1	10	8
	10	-1	