Examen de Programación 1
Instituto de Computación - Facultad de Ingeniería
Diciembre 2016

Leer con atención

- Todos los códigos deben ser escritos en el lenguaje **Pascal** tal como fue dado en el curso. A grandes rasgos, este es el Pascal estándar con los siguientes agregados: utilización de `else` en la instrucción `case` y evaluación por circuito corto de las operaciones booleanas (`and` y `or`).
- 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. De esta manera se restarán puntos entre otros conceptos por: mala o nula indentación, mala utilización de las estructuras de control, código confuso e innecesariamente largo, programas ineficientes, utilización de variables globales, pasaje incorrecto de parámetros, etc. No obstante, por razones prácticas no exigimos que incluya comentarios en los códigos que escriba en la prueba.
- **Escriba su nombre completo y cédula en todas las hojas. Numere todas las hojas y escriba la cantidad total de hojas.**
- **Escriba de un solo lado de la hoja y comience cada ejercicio en una nueva hoja**.

Ejercicio 1

Dada la siguiente definición de tipos, con una constante `N` dada:

```
type natural = 0 .. maxint;
     arreglo = array [1..N] of natural;
```

Se pide implementar la siguiente función:

```
function sumaConsecutivas (a : arreglo; suma : natural ) : boolean;
```

que, dado un arreglo, indica si existe una secuencia de celdas consecutivas dentro del arreglo cuya suma es igual a un valor dado. Ejemplos para `N = 6`:

- `suma = 5` y `a = 1 2 3 2 7 2`, retorna `TRUE` (`a[2] + a[3]` = `5`)
- `suma = 12` y `a = 1 2 3 2 7 2`, retorna `TRUE` (`a[3] + a[4] + a[5]` = `12`)
- `suma = 10` y `a = 1 2 3 2 7 2`, retorna `FALSE` (no existen celdas consecutivas cuya suma dé `10`)

Ejercicio 2

Se considera la siguiente definición de tipos, con `N` y `natural` los del ejercicio anterior:

```
type ArregloConTope = record
                        arre : array [1..N] of natural;
                        tope : 0..N;
                      end;
     Resultado = record
                   case encontrado : boolean of
                       true  : (cantidad : 1..N);
                       false : ();
                 end;
```

Escribir el siguiente procedimiento:

```
procedure maximoMenorQue (valor : natural; a : ArregloConTope; var res : Resultado);
```

que busca en el arreglo con tope `a` el mayor elemento que sea menor que `valor` y retorna en el campo `cantidad` de `res` la cantidad de veces que se encuentra en `a`. De no haber ningún elemento que cumpla la condición, el campo `encontrado` de `res` queda en `false`. Considere que `res` viene indefinido. Ejemplos para `N = 10`:

valor	a	res
24	[7,21,22,321,22]	encontrado = true, cantidad = 2
7	[7,21,22,321,22]	encontrado = false
5	[ ]	encontrado = false

Ejercicio 3

Se considera la siguiente definición de tipos:

```
type  ListaChar = ^Celda;
      Celda = record
         elem : char;
         sig  : ListaChar;
      end;
```

Escribir el siguiente procedimiento:

```
procedure cortarEn (largo : integer; var l, res : ListaChar);
```

que corta la lista `l` de entrada para que tenga (como máximo) `largo` elementos y retorna en `res` el resto. Considere que `res` viene indefinida (no apunta a `nil` ni tiene memoria asignada). Ejemplos:

largo	l inicial	l modificada	res modificada
2	['a','b','c','d']	['a','b']	['c','d']
0	['a','b','c','d']	[ ]	['a','b','c','d']
10	['a','b','c','d']	['a','b','c','d']	[ ]
10	[ ]	[ ]	[ ]

Ejercicio 4

Indique cuál es la salida del siguiente programa si se ingresa como entrada el cuarto dígito comenzando desde la izquierda de su cédula de identidad. Así por ejemplo, si su cédula es **4567890-1**, se ingresará **7**.

```
program ejemplo;
var x,y,z : integer;
procedure de(x: integer; var z: integer);
begin
   x := x + y - z;
   z := x - 1
end;
function muy(x, y : integer) : integer;
var z : integer;
   function mal(x : integer) :  integer;
   begin
      x   := x - z;
      mal := x
   end;
begin
   z := x + y;
   y := mal(y);
   writeln(y);
   muy := x
end;
procedure programa(var x : integer; y : integer);
begin
   y := y + muy(x,z);
   de(y,x);
   writeln(y)
end;
begin
   readln(x);
   y := x + 1;
   z := x - 1;
   programa(z, x);
   writeln(z);
   writeln(x)
end.
```