Examen - Programación 1
Febrero 2022

Ejercicio 1 (30 puntos)

Dada la siguiente declaración de una lista de enteros:

```
type lista = ^info;
     info  = record
               val: integer;
               sig: lista
             end;
```

Definir el procedimiento:

```
procedure generarListaHasta (k:integer; var l:lista);
```

que dado un entero `k` genera una lista de enteros `l` tal que la suma de sus elementos no supera `k`. Los elementos de la lista están formados por la sucesión `F(0)`, `F(1)`, `F(2)`, ..., donde `F` es una función Pascal dada que devuelve enteros estrictamente mayores que cero. Se asume que `k >= 0`.

Ejemplos: Supongamos que `F(i)` retorna `2 * i + 1`. Entonces:

entrada `k`	salida `l`
10	(1,3,5)
9	(1,3,5)
0	`NIL`

Nota: El orden de los elementos de la lista generada es irrelevante. Por lo tanto, por ejemplo, la siguiente lista también es válida:

entrada `k`	salida `l`
10	(5,3,1)

Ejercicio 2 (20 puntos)

El triángulo de Pascal es un triángulo cuyo renglón k-ésimo contiene k enteros. El primer renglón tiene un 1. Los siguientes renglones se forman a partir del anterior de forma que la primera y la última celda son un uno, y la celda i+1 es la suma de los contenidos de las celdas i e i+1 del renglón anterior. Por ejemplo, este es el triángulo de Pascal con cinco renglones.

```
      1
     1 1
    1 2 1
   1 3 3 1
  1 4 6 4 1
```

Consideremos la siguiente declaración de tipos:

```
const MAX = ...;    { MAX > 1 }
type  RTP = record { Renglon del Triangulo de Pascal }
              info: array[1..MAX] of integer;
              tope: 1..MAX;
            end;
      TP  = array [1..MAX] of RTP;  { Triangulo de Pascal }
```

Definir el procedimiento:

```
procedure construirTP (var t:TP);
```

que devuelve el triángulo de Pascal de MAX renglones.

Ejercicio 3 (20 puntos)

Consideremos la siguiente declaración de tipo para representar palabras de largo `LPAL`:

```
const LPAL = ...; { LPAL >= 1 }
type  palabra = array[1..LPAL] of char;
```

Definir la función:

```
function casiIgual (p1, p2: palabra): boolean;
```

que devuelve `true` si y solamente si las palabras `p1` y `p2` tienen exactamente una letra diferente.

Ejemplos:

p1	p2	casiIgual
PASTA	PISTA	true
PASTA	POSTE	false
PASTA	PASTA	false

Ejercicio 4 (30 puntos)

El laboratorio de robótica cuenta con diferentes tipos de robots educativos, algunos diseñados en facultad (Butiá y Robotito) y otros que son comerciales. Se cuenta con el stock de los robots disponibles, donde para cada uno se tiene un identificador **único** y diferente información según el tipo. Si un robot es del tipo Robotito, se tiene el color; si es un Butiá, la cantidad de sensores; y si es comercial la cantidad de sensores y su versión.

Consideremos la siguiente declaración de tipos:

```
const N = ...; { N >= 1 }
type TColor = (Anaranjado, Amarillo, Azul);
     TRobot = (Butia, Comercial, Robotito);
     InfoRobot = record
                   Id : Integer;
                   case Robot : TRobot of
                     Butia     : (CantSensoresB : integer);
                     Comercial : (CantSensoresC,
                                  Version : integer);
                     Robotito  : (Color : TColor)
                 end;
     TStock = record
                stock : array [1..N] of InfoRobot;
                tope  : 0..N
              end;
```

a) **(15 puntos)** Definir el procedimiento:

```
procedure eliminarRobot(var s:TStock; id:integer);
```

que dado un identificador, elimina el correspondiente robot del stock. El procedimiento deja el stock como estaba si no existe un robot con dicho identificador.
Nota: El orden de los robots en el stock es irrelevante, por lo que al salir de este procedimiento no interesa si algunos robots quedan en otro orden.

b) **(15 puntos)** Definir la función:

```
function cantidadSensores(s:TStock):integer;
```

que retorna el total de sensores que hay en el stock.