Programación 1
Segundo Parcial
Instituto de Computación - Facultad de Ingeniería
Julio 2019

Leer con atención

- Todos los programas o fragmentos de programas deben ser escritos en el lenguaje **Pascal** tal como fue dado en el curso. A grandes rasgos, este es el Pascal estándar con algunos agregados, a saber:
  - Utilización de `else` en la instrucción `case`.
  - 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

Considere las siguientes declaraciones:

```
CONST
   CANT_PERS = ...; {entero mayor que 0}
   MAX_CADENA = ...; {entero mayor o igual que 0}

TYPE
   Cadena = RECORD
      letras : ARRAY [1..MAX_CADENA] OF Char;
      largo : 0..MAX_CADENA;
   END;

   Enumerado = (menor, adulto, nadie);

   Persona = RECORD
      nombre : Cadena;
      case generacion: Enumerado of
         menor : (edad : 0..20);
         adulto : (salario : Real);
         nadie : ();
   END;

   Grupo = ARRAY [1..CANT_PERS] OF Persona;
```

Parte a)

Escribir la función:

```
      function cadenasIguales (cad1, cad2 : Cadena) : Boolean;
```

que, dadas dos cadenas, devuelve *True* si son iguales y *False* si no.

Parte b)

Escribir el procedimiento:

```
procedure buscarPersonaUnica (nombre: Cadena; arreglo : Grupo; var personaEncontrada : Persona;
                                                                var  esUnica :Boolean);
```

que, dados un nombre y una arreglo de personas, devuelve los siguientes resultados:

- en el parámetro *personaEncontrada*, se devuelve la primera persona del arreglo cuyo nombre coincide con el nombre dado. Si no hay ninguna persona con el nombre dado, en el campo discriminante se devuelve *nadie* y en el campo *nombre* se devuelve una cadena vacía.
- en el parámetro *esUnica*, se devuelve *TRUE* si no existen más personas con ese nombre, y *FALSE* en cualquier otro caso.

Parte c)

Escribir el procedimiento:

```
procedure maximos (arreglo : Grupo; var maxSalario : Real; var maxEdad : Integer);
```

que, dado una arreglo de personas, devuelve en el parámetro *maxSalario* el mayor salario y en el parámetro *maxEdad* la mayor edad. Si no hay adultos devuelve -1.0 en *maxSalario* y si no hay menores devuelve -1 en *maxEdad*.

Ejercicio 2

Dadas las siguientes declaraciones::

```
TYPE
   ListaInt = ^Celda;
   Celda = RECORD
      dato: Integer;
      sig: ListaInt
   END;
```

Escribir el procedimiento:

```
prodedure insertar (var lis1, lis2 : ListaInt; pos : Integer);
```

que, dadas dos listas *lis1* y *lis2* distintas de *NIL* y una posición *pos* mayor que 0, inserta *lis1* después de la posición *pos* de *lis2*. Los elementos de *lis2* a partir de *pos* quedan luego de los insertados. Si *pos* es mayor que el largo de *lis2*, se inserta *lis1* al final de *lis2*.

Ejemplos:

Si *lis1* es [1,2,3], *lis2* es [5,6,7,8] y *pos* es 2, el resultado de *insertar(lis1, lis2, pos)* es [5,6,1,2,3,7,8]
Si *lis1* es [1,2,3,4], *lis2* es [5,6,7] y *pos* es 5, el resultado de *insertar(lis1, lis2, pos)* es [5,6,7,1,2,3,4]

Se supone ya implementada la siguiente función, que puede utilizar en el procedimiento solicitado:

```
function ultimo (lis : ListaInt) : ListaInt;
```

La función *ultimo*, dada una lista *lis* distinta de *NIL*, devuelve un puntero a su última celda.

Ejercicio 3

Sea la secuencia $S=A_0, A_1, A_2, \ldots$, en donde $A_i=1/2^i$, o sea, $S=1, 1/2, 1/4, \ldots$

Escribir la función:

```
function indicePrimerMenor(r : Real): Integer;
```

que, dado un valor Real `r`, tal que $0 < r <= 1$, devuelve el índice del primer elemento de la secuencia S que es menor a `r`.

Ejemplos:

r	resultado	comentarios
1.0000	1	$A_0=1$ no es menor que *r* , $A_1=1/2$ sí es menor que *r*
0.5001	1	$A_0=1$ no es menor que *r* , $A_1=1/2$ sí es menor que *r*
0.5000	2	$A_1=1/2$ no es menor que r , $A_2=1/4$ sí es menor que r