Programación 1 - Segundo Parcial
Instituto de Computación - Facultad de Ingeniería
Julio 2022

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 (42 pts)

Dados los siguientes tipos de datos:

```
CONST
  MaxDigito = ...;      (* constantes de valores adecuados *)
  MaxUsuarios = ...;

TYPE
  Documento = ARRAY [1..MaxDigito] of 0..9;
  Estado = (Agendado, Atendido, Ausente);
  Fecha = RECORD
             dia : 1..31;
             mes : 1..12;
             anio : 1930..2030;
          END;

  Usuario  = RECORD
                 documento : Documento;
                 CASE est : Estado of
                     Atendido : (fechaAtendido : Fecha);
                     Agendado : (fechaAsignada : Fecha);
                     Ausente : ();
              END;

  Agenda = RECORD
               usuarios : ARRAY [1..MaxUsuarios] OF Usuario;
               tope : 0..MaxUsuarios;
           END;
```

Parte a (12 pts)

Escribir la función *docIguales* que, dados dos números de documento, devuelve true si y solo si son iguales, false si no.

```
      function docIguales (doc1, doc2: Documento): boolean;
```

Parte b (6 pts)

Escribir la función *fechasIguales* que, dadas dos fechas, devuelve true si y solo si son iguales, false si no.

```
      function fechasIguales (fec1, fec2: Fecha): boolean;
```

Parte c (12 pts)

Escribir la función *estadoUsuario* que, dados un número de documento y una agenda, devuelve el estado del usuario de la agenda con ese número de documento, en caso de que exista en la agenda (se supone que está a lo sumo una vez). En caso de no existir, devuelve `Ausente`.

```
function estadoUsuario (doc: Documento; age: Agenda): Estado;
```

Parte d (12 pts)

Escribir el procedimiento *usuariosFecha* que, dados una fecha y una agenda, devuelve en el parámetro `atendidos` todos aquellos usuarios de la agenda que fueron atendidos en esa fecha (no importa en qué orden).

```
procedure usuariosFecha (fec: Fecha; age: Agenda; var atendidos: Agenda);
```

Ejercicio 2 (18 pts)

Dados los siguientes tipos de datos:

```
TYPE
  Lista = ^Nodo;
  Nodo = RECORD
            elem : Char;
            sig : Lista;
         END;
```

Parte a (6 pts)

Escribir el procedimiento *agregaPrincipio* que, dados un carácter y una lista, agrega un nuevo nodo con dicho carácter al principio de la lista.

```
procedure agregaPrincipio (elem: char; var list: Lista);
```

Parte b (12 pts)

Escribir el procedimiento *soloAlfaNumericos* que, dada una lista `list`, devuelve en el parámetro `resultado` otra lista con los elementos alfanuméricos de la lista `list` (en orden **inverso** al que aparecen en la lista `list`). Puede suponer ya implementada la función `function esAlfaNumerico (elem: char): boolean;` que, dado un carácter devuelve true si es alfanumérico, false si no lo es.

```
procedure soloAlfaNumericos (list: Lista; var resultado: Lista);
```

Ejemplo: Para `list = ['F','#','2','s','&']` devuelve `resultado = ['s','2','F']`