Programación II

Lic. en Sistemas de Información

Evaluación Parcial

Alumno: Apellido y Nombre *Leg:* Número **12 / 07 / 2021**

Aclaración:
En todos los ejercicios que se pide un algoritmo se debe realizar el estudio de la complejidad del mismo

---

1) Escribir una función que dada una lista **L**, agrupa de a **n** elementos dejando su suma.

Function junta(L: Lista, n: integer): Lista
Nota: No usar ninguna estructura auxiliar.

Dada la lista **L := [1, 3, 2, 4, 5, 2, 2, 3, 4, 7, 4, 3, 2, 2]**
junta(L,3) retorna **[6, 11, 9, 14, 4]**

1 + 3 + 2 = 6
4 + 5 + 2 = 11
2 + 3 + 4 = 9
7 + 4 + 3 = 14
2 + 2 = 4

```
Function junta(L: Lista; n: integer): Lista;
var I: PosicionLista;
  X, Y: TipoElemento;
  Nulo: TipoElemento;
  Suma: TipoElemento;
  Lista_Suma: Lista;
  o: Integer;
begin
  Lista_Suma.Crear(Numero, L.SizeList);
  I := L.Comienzo;
  while (I < L.Fin) and (I <> 0) do
  begin
    X.Clave := 0;
    o := 1;
    while o <= n do
    begin
      Y := L.Recuperar(I);
      X.Clave := X.Clave + Y.Clave;
      o := o + 1;
      I := L.Siguiente(I);
    end;
    Suma.Clave := X.Clave;
    Lista_Suma.Agregar(Suma);
  end;
  Result := Lista_Suma;
end;
```

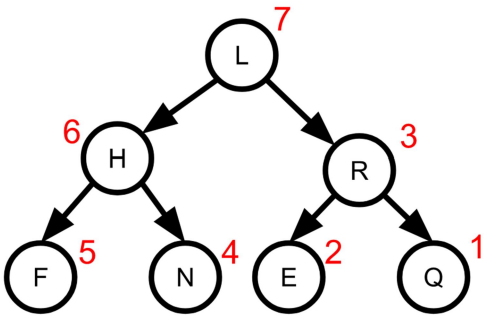
Se considerará regular

Esto no funciona para apuntadores porque no es posible la comparación "i <> 0", directamente NO compila.

REGULAR. Solo funcionará con lista con arreglos porque cuando quiera comparar "I: PosicionLista" contrá un "0" en "ListPointer" directamente NO compila. Además siempre se recomienda recorrer la lista como "While (I <> Nulo)" eso si es transparente a todas las implementaciones ya que "I" y "Nulo" están definidos del mismo tipo siempre, sin importar la implementación.

---

2) Dado un árbol binario escribir una función ***iterativa*** que retorne una lista con las claves en el siguiente orden:



Function recorridoArbol(Var A:Arbol): Lista

Resultado esperado: una lista con los elementos -> **[Q, E, R, N, F, H, L]**

```
Function recorridoArbol(var Tree:Arbol): Lista;
var
  X: TipoElemento;
  List: Lista;
  Procedure AUXrecorridoArbol(P: PosicionArbol);
  begin
    If Tree.RamaNula(P) Then
    begin
    end
    Else
    Begin
      AUXrecorridoArbol(P^.HD);
      AUXrecorridoArbol(P^.HI);
      List.Agregar(P^.Datos);
    End;
  End;
Begin
  List.Crear(Cadena,Tamaño);
  AUXrecorridoArbol(Tree.Root);
  recorridoArbol := List;
End;
```

Se considerará incorrecto

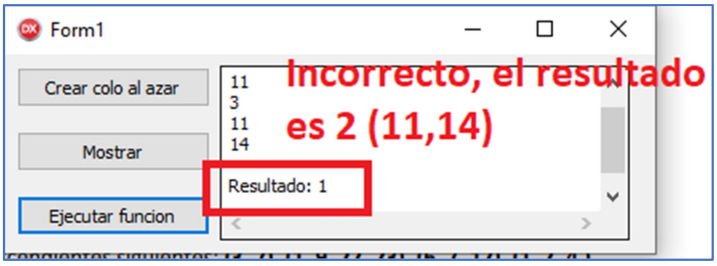
Esto no es iterativo como pide el enunciado

INCORRECTO

---

3) Se tiene una **cola** con números enteros positivos desordenados, se pide encontrar la longitud de la secuencia de ascendientes más grande que existe.
Por ejemplo, si se tiene la siguiente secuencia: **3, 7, 1, 9, 22, 23, 6, 7, 12, 1, 2, 4, 2.**
Se tienen las secuencias ascendientes siguientes: **(3, 7), (1, 9, 22, 23), (6, 7, 12), (1, 2, 4,), (2)**, la respuesta debiera ser 4.

Escribir una función que dada una cola retorne un entero con la longitud máxima correspondiente.



INCORRECTO

---

4) Se desea implementar un sistema para la Cámara Electoral de la Nación para el registro de votos. El sistema debe poder registrar los votos en cada mesa. Cada mesa corresponde a un distrito y ese distrito a una provincia. Son 24 provincias y, en cada provincia los distritos se numeran de 1 a 50. Y en cada distrito las mesas se numeran nuevamente de 1 a 1000. Por cada mesa el sistema registra:

- el número de mesa, el distrito y la provincia.
- cada candidato con la cantidad de votos que sacó en esa mesa.

La mesa 230 del distrito 4 de la provincia de Bs As tiene 150 votos para MM y 200 para AF. Se desea, luego de terminada la carga, poder responder las siguientes consultas:

- Listar cada candidato junto a su conteo de votos totales (entre todas las mesas del país).
- Dada una mesa, ver cuántos emitieron su voto en la misma.

a) Proponer una estructura de datos (puede ser una sola o una combinación de las estructuras vistas en la materia) a utilizar para guardar los datos de manera que la inserción sea rápida y las búsquedas sean lo más eficiente posibles y justificar.
b) Determinar la complejidad algorítmica de una inserción y justificar.
c) Para la estructura definida, determinar la complejidad algorítmica de cada una de las consultas solicitadas y justificar.

INCORRECTO. Se pedía especificar que estructura utilizar y no hacer el código. Más allá de eso el código realizado utilizando listas será lento ya que todas las operaciones serán de orden "N", lo que se pide es utilizar alguna estructura que mejore ese orden como un árbol AVL o una tabla hash.

NOTA DEL PARCIAL = **DESAPRODADO**