Examen, forma 1

Aprendizajes a evaluar:

- *Define un tipo de dato abstracto (TDA) para requerimiento computacional utilizando estándares de diseño de programación*
- *Construir un programa computacional para resolver un problema de usuario utilizando estructuras de datos estáticas tales como arreglos, pilas y colas con Lenguaje Programación C.*
- *Utiliza la recursividad con el fin de generar programas eficientes a través de estándares de programación.*
- *Construir un programa computacional para resolver un requerimiento de un usuario utilizando estructuras de datos dinámicas-listas.*
- *Construir un programa computacional para resolver un requerimiento de un usuario utilizando estructuras de datos dinámicas-árboles.*

Instrucciones:

Estimados y estimadas estudiantes,
Bienvenidos a esta actividad denominada "Evaluación Examen", de la asignatura Estructura de Datos. A continuación se indican las instrucciones generales que debes considerar para que desarrolles esta evaluación sin inconvenientes.

- Dispones de 180 minutos para responder, se informará en pantalla el tiempo restante una vez iniciada.
- Dispones de un solo intento para la evaluación, esto significa que si la inicias debes terminarla.
- Los programas debe escribirlos en algún entorno de desarrollo (IDE) para Lenguajes de Programación C, indicando claramente en cada programa, la pregunta que está contestando.
- Debe crear una carpeta con su ***nombre y apellido*** que la utilizará para guardar los programas que usted creará para responder esta evaluación.
- Cuando termine de responder esta evaluación, comprime en formato zip o rar la carpeta que tiene su nombre y súbela al foro destinado para esto. Debes subirla antes que acabe el tiempo que dispones para la evaluación.
- Esta evaluación corresponde al 30% de tu nota de la asignatura.
- Esta evaluación estará disponible sólo en la semana 17.
- La nota se calcula con una escala de exigencia al 60% (60% del PT=4.0)
- La puntuación total de esta evaluación es de 100 puntos (30, 40 y 30 puntos de cada parte).
- Los programas y respuestas de esta evaluación debe ser de su autoría, en caso contrario será considerado como copia o plagio; y su evaluación será un 1.
- Esta evaluación debe desarrollarla individualmente. No puede solicitar apoyo.
- El resultado y la pauta de corrección se liberarán el lunes de la semana 18 a las 00:05 hrs.

Cualquier duda, inquietud o consulta que puedes tener antes de la evaluación, puedes realizarla en el Foro de Consultas.

¡¡¡Éxito en la actividad!!!

EXAMEN

CASO 1 [30 puntos]:

Instrucciones:

- Dentro de la carpeta que tiene su nombre, cree una subcarpeta llamada CASO1 y guarde los programas que creará para responder este caso.
- Para responder este caso, debe programar en lenguaje C y utilizar y adecuar el TDA elemental de listas que usted implementó durante la unidad del curso o durante esta evaluación (no puede utilizar uno que descargue o copie de otras fuentes ya que será considerado como plagio). El TDA elemental debe adecuarlo a la estructura siguiente:

```
typedef struct info{
  int valor;
}Info;
typedef struct nodo{
  Info *dato;
  struct nodo *sgte;
}Nodo;
typedef struct lista{
  Nodo *ini;
  Nodo *fin;
  int tam;
}Lista;
```

- Implemente un nuevo programa en lenguaje C que incluya como librería (o Header File) el TAD anterior y agregue las funciones que se solicitan a continuación.uerde que su programa debe tener la estructura completa de un programa C.

Descripción del Caso:

1) Implemente la función ***recursiva*** cuentaNodos que reciba por parámetros una lista de enteros L, y que encuentre y retorne la cantidad de nodos de la lista. Si la lista L no tiene nodos , la función debe retornar un 0.
   Ejemplos:
   Para L: [1,2,6,5,8,3,4,6] debe retornar 8
   Para L: [] debe retornar 0

2) Implemente la función sumaSublista que reciba por parámetros una lista de enteros L y un entero S, y que encuentre y retorne una sublista cuya suma sea S. Si no existe ninguna sublista con dicha suma, retorne una lista vacía. En caso de haber varias listas que cumplan retorne la primera que encuentra.
   Ejemplos:
   Para L: [1,2,6,5,8,3,4,6] y S=13 debe retornar [2,6,5]
   Para L: [1,2,6,5,8,3,4,6] y S=15 debe retornar [8,3,4]
   Para L: [1,2,6,5,8,3,4,6] y S=17 debe retornar []
   Para L: [1,2,6,5,8,3,4,6] y S=20 debe retornar [5,8,3,4]

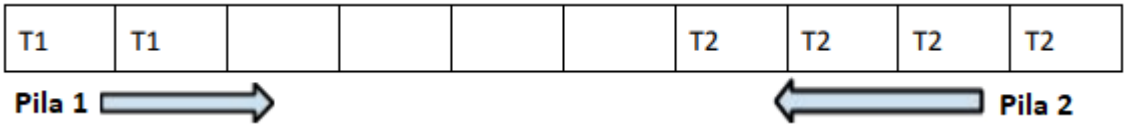
CASO 2 [35 puntos]:

Instrucciones:

- Dentro de la carpeta que tiene su nombre, cree una subcarpeta llamada CASO2 y guarde el documento y programas que creará para responder este caso.
- Para responder este caso primero debe especificar un TDA según las restricciones que describe este caso. Esta especificación escríbalo en un documento Word.
- Implemente el TDA especificado usando Lenguaje de programación C. Recuerde que su programa debe tener la estructura completa de un programa C.

Descripción del Caso:

1) Es necesario describir una especificación de un ***TDA PilaDoble estática*** de elementos de un tipo genérico T que considere un conjunto mínimo de operaciones. Una PilaDoble estática es una estructura que representa 2 pilas estáticas que ocupan una misma estructura, donde una de las pilas crece de izquierda a derecha y la otra crece de derecha a izquierda.



   Su especificación debe realizarla utilizando descripción Informal y Formal y debe considerar las siguientes operaciones cómo mínimo para cada pila:
   - Agregar un elemento
   - Quitar un elemento
   - Que indique si la pila está llena
   - Mostrar el tope

2) Implemente el TDA especificado anteriormente considerando un main de prueba cuyos elementos agréguelos utilizando números aleatorios (random). Su prueba debe considerar la ejecución de todas las operaciones elementales para ambas pilas.

CASO 3 [35 puntos]:

Instrucciones:

- Dentro de la carpeta que tiene su nombre, cree una subcarpeta llamada CASO3 y guarde los programas que creará para responder este caso.
- Para responder este caso, debe programar en lenguaje C y utilizar y adecuar el TDA elemental de Árboles Binarios de Búsqueda que usted implementó durante la unidad 6 del curso o durante esta evaluación (no puede utilizar uno que descargue o copie de otras fuentes ya que será considerado como plagio). El TDA elemental debe tener la estructura siguiente:

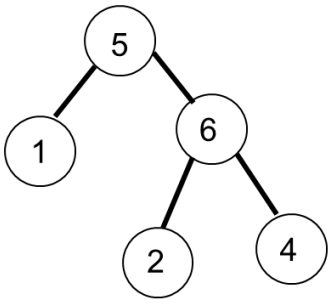
```
typedef struct info {
  int clave;
}Info;
typedef struct nodoArbol {
    Info *info;
    struct nodoArbol *izq;
    struct nodoArbol *der;
}Nodo;
typedef struct arb{
    Nodo *raiz;
}Arbol;
```

- Implemente un nuevo programa en lenguaje C que incluya como librería (o Header File) el TDA anterior y agregue las funciones que se solicitan a continuación. Recuerde que su programa debe tener la estructura completa de un programa C.

Descripción del Caso:

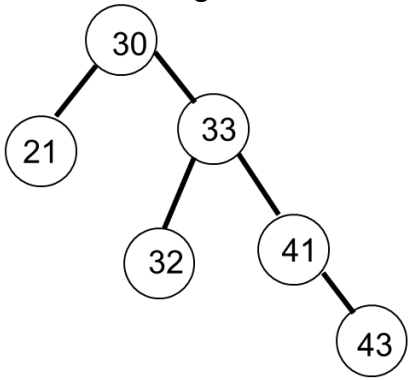
1) Dado un árbol binario A, generar la función ***esABB*** que devuelve true si A es un árbol binario de búsqueda y false en caso de que no lo sea. Recordar que un árbol binario es un ABB si para A distinto de vacío, cualquier nodo n del árbol, todos los valores del subárbol izquierdo del nodo n son menores al valor de n. Para cualquier nodo n del árbol, todos los valores del subárbol derecho del nodo n son mayores al valor de n.
   Ejemplos:

Para el árbol siguiente:



Debe retornar false

Para el árbol siguiente:

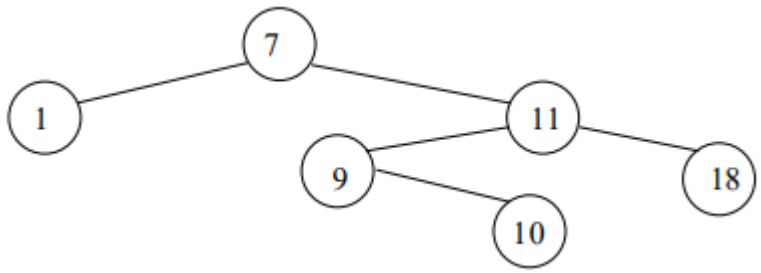


Debe retornar true

2) Crear un árbol R, que dado un árbol A de tipo ABB, devuelve otro árbol estructuralmente igual a A que contiene en cada nodo la suma del mayor elemento del subárbol izquierdo con el menor elemento del subárbol de la derecha (Max(**izq**) + Min(**der**))

Ejemplo:

Si tiene el árbol siguiente:



El resultado debe ser: