

**Pregunta 1**

Finalizado

**Cuales son las diferencias entre recursividad e iteración con listas****Pregunta 2**

Finalizado

Puntúa como

**Pregunta 3**

Parcialmente correcta

Se puntúa 3.00 sobre 5.00

▼ Marcar pregunta

Hacer una función para pasar los elementos de un abb a un archivo, la raíz del árbol tiene que quedar al final del archivo.  
Considere que el árbol es de enteros.

Una el prototipo de las funciones con una sinopsis de su funcionamiento

void mostrarFila(Referencia\* fila);

Recorre la fila con un for

int filaVacia(Referencia\* fila);

Retorna 0 o 1 si la fila no contiene elementos.

void inicReferencia(Referencia\* fila);

Asigna null al parámetro enviado para inicializar la fila

int extraer(Referencia\* fila);

Retorna y elimina el primer elemento de la fila

void agregar(Referencia\* fila, nodo2\* nuevo);

Añade un nuevo nodo a la fila y actualiza las referencias.

Se le pide a un alumno que haga una función que busque el menor elemento de una lista de manera recursiva. El alumno escribió esta función:

```
typedef struct{
    int nro;
    struct nodo* sig;
}nodo;

nodo* buscarMenorLista(nodo* lista) {
    nodo* menor = lista->nro;
    if(lista != NULL){
        menor = buscarMenorLista(lista->sig);
    }else{
        if(menor > lista->nro)
            menor = lista;
    }
    return menor;
}
```

Determinar si la solución es correcta o no. En caso que sea correcta proporcionar si lo hay una mejor solución y en caso que sea incorrecta marcar los errores.

**Pregunta 6**

Finalizado

Puntúa como

10.00

▼ Marcar pregunta

Dadas las siguientes estructuras de datos:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 typedef struct{
5     int id;
6     char apellido[30];
7     char nombre[30];
8     char dni[10];
9     char cuit[15];
10    char email[90];
11    char domicilioParticular[50];
12    char domicilioComercial[50];
13    char telefonoFijo[15];
14    char telefonoCel[15];
15    int cargo;
16 }stCliente;
17
18 typedef struct{
19     stCliente dato;
20     struct nodoLista *sig;
21 }nodoLista;
22
23 typedef struct{
24     stCliente dato;
25     struct nodoArbol *izq;
26     struct nodoArbol *der;
27 }nodoArbol;
28
29 stCliente cargaUnCliente();
30 void muestraUnCliente(stCliente c);
```

Hacer una función que recorra una lista y copie su contenido a un árbol binario.

La función recibe la lista por parámetro (ya cargada con datos) y retorna el árbol.

**Modularizar y desarrollar las funciones que necesite para crear y administrar el árbol binario. El árbol binario debe estar ordenado por dni.**

Suponga la existencia de un archivo de datos binarios que almacena personas registradas como dadores de sangre. La estructura tiene la siguiente forma:

```
typedef struct
{
    int dni;
    char nombre[30];
    char apellido[30];
    char telefono[30];
    int edad;
    int idSangre;
    char grupoSanguineo; // A, B, AB, O
    char factorRh; // + o -
} registroArchivo;
```

1. Diseñe la estructura compuesta que permita organizar la información de manera dinámica, detallando las estructuras de datos resultantes.
2. Detalle las funciones básicas que necesita para administrar cada una (prototipado) y explique brevemente el funcionamiento de cada una.
3. Justifique el porqué de la estructura compuesta elegida.

Pregunta 7

Correcta

Se puntuó 5.00 sobre 5.00

▼ Marcar pregunta

Dos de las cuatro reglas de la buena recursividad son:

Seleccione una o más de una:

- 1. El operador ternario
- 2. Condición de corte
- 3. Al llegar a la Solución Trivial queda expresada la solución total
- 4. El uso de la instrucción if

✓ Si es una de las 4 reglas

✓ Si, es la cuarta regla de la buena recursividad

Respuesta correcta

Las respuestas correctas son: Condición de corte, Al llegar a la Solución Trivial queda expresada la solución total

Pregunta 11

Finalizado

Puntúa como 10,00

▼ Marcar

Responda con sus palabras:

1. ¿Qué es un árbol binario de búsqueda?
2. ¿Con qué criterio se insertan los nuevos elementos en un árbol binario?
3. ¿Qué beneficio nos brinda utilizar este tipo de estructura de datos?

Pregunta 10

Parcialmente correcta

Se puntuó 4,11 sobre 5,00

▼ Marcar pregunta

Vincular cada estructura con su correspondiente característica

En esta estructura cada nodo almacena la dirección del nodo siguiente y del nodo anterior.

Lista doblemente enlazada



Se necesitan dos vínculos en su estructura, uno para enlazar nuevos elementos y otro para quitarlos.

Lista simplemente enlazada



Invierte el orden de los elementos que en ella se almacenan.

Pila



Los datos se deben insertar en el medio para quitarlos más fácilmente.

No coincide con ninguna estructura aprendida



No se permite acceder a elementos intermedios.

Pila y Fila



El elemento que se almacenó primero es el primero en removérse.

Fila



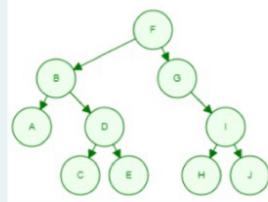
Respuesta parcialmente correcta.

Ha seleccionado correctamente 5.

La respuesta correcta es: En esta estructura cada nodo almacena la dirección del nodo siguiente y del nodo anterior. → Lista doblemente enlazada. Se necesitan dos vínculos en su estructura, uno para enlazar nuevos elementos y otro para quitarlos. → Fila. Invierte el orden de los elementos que en ella se almacenan. → Pila. Los datos se deben insertar en el medio para quitarlos más fácilmente. → No coincide con ninguna estructura aprendida. No se permite acceder a elementos intermedios. → Pila y Fila. El elemento que se almacenó primero es el primero en removérse. → Fila

Pregunta 8  
Correcta  
Se puntuá 10,00 sobre 10,00  
 Marcar pregunta

Analice el siguiente árbol y responda:



\* Altura total del árbol:  ✓

\* Grado del nodo A:  ✓

\* Si elimino el Nodo B y elijo ir por la derecha, entonces B se reemplaza por:  ✓

\* Si se me muestra el árbol de esta manera: A C E D B H J I G F, significa que se está recorriendo en:  ✓

Pregunta 9  
Correcta  
Se puntuá 5,00 sobre 5,00  
 Marcar pregunta

Arboles binarios, vincular cada característica con su definición:

Nodo hoja	Es aquél que tiene padre pero no tiene hijos	<input type="checkbox"/>	✓
Nodo rama	Es aquél que tiene padre e hijos	<input type="checkbox"/>	✓
Grado	Es la descendencia de cada nodo	<input type="checkbox"/>	✓
Nivel	Aumenta a medida que se aleja de la raíz	<input type="checkbox"/>	✓
Altura	Parte de la raíz y depende de su descendencia	<input type="checkbox"/>	✓
Nodo raíz	Es aquél que tiene hijos pero no tiene padre	<input type="checkbox"/>	✓

Respuesta correcta

La respuesta correcta es: Nodo hoja → Es aquél que tiene padre pero no tiene hijos, Nodo rama → Es aquél que tiene padre e hijos, Grado → Es la descendencia de cada nodo, Nivel → Aumenta a medida que se aleja de la raíz, Altura → Parte de la raíz y depende de su descendencia, Nodo raíz → Es aquél que tiene hijos pero no tiene padre

Pregunta 6  
Finalizado  
Puntúa como 10,00  
 Marcar pregunta

Dadas las siguientes estructuras de datos:

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  typedef struct{
5      int id;
6      char apellido[30];
7      char nombre[30];
8      char dni[10];
9      char cuit[15];
10     char email[90];
11     char domicilioParticular[50];
12     char domicilioComercial[50];
13     char telefonoFijo[15];
14     char telefonoCel[15];
15     int cargo;
16 }stCliente;
17
18 typedef struct{
19     stCliente dato;
20     struct nodoLista *sig;
21 }nodoLista;
22
23 typedef struct{
24     stCliente dato;
25     struct nodoArbol *izq;
26     struct nodoArbol *der;
27 }nodoArbol;
28
29 stCliente cargaUnCliente();
30 void muestraUnCliente(stCliente c);
  
```

Hacer una función que recorra una lista y copie su contenido a un árbol binario.

La función recibe la lista por parámetro (ya cargada con datos) y retorna el árbol.

**Modularizar y desarrollar las funciones que necesite para crear y administrar el árbol binario. El árbol binario debe estar ordenado por dni.**