SIMULACRO PARCIAL PRG 2

1. ¿Cuando es recomendable utilizar recursividad en lugar de iteracion? ¿Cuales son las desventajas de utilizar recursividad en lugar de iteracion?

2. Escribir en pseudocodigo los pasos necesarios para crear un nodo en una lista lista doblemente enlazada

3. Definir en pseudocodigo los pasos a seguir para borrar un nodo de una lista simple

4. ¿Que es una lista doble y que ventaja tiene sobre una lista simple?

5. ¿Que es un puntero?

    a.Un puntero es una dirección de memoria asociada a un identificador (nombre) y a un tipo de dato, que almacena una dirección de memoria de un dato (variable) del mismo tipo del puntero.

    b.Un puntero es un tipo de dato que nos ofrece la solución de queremos repetir una tarea un número determinado de veces. Para esto utiliza generalmente variables auxiliares, como un contador y posee una condicion de comprobacion.

    c.Un puntero es lo que llamamos en forma genérica como "módulo" en el lenguaje C, admiten argumentos, que son datos que le pasan como valor o referencia y pueden devolver datos usando la palabra return.

    d.Un tipo de variable que una vez inicializada no puede cambiar su valor en ningun momento del programa. Generalmente se escriben en mayusculas.

6. Marque las opciones que sean correctas para una lista simple

    a.A cada uno de los elementos de la lista vinculada lo llamamos DATO

    b.Se unen al siguiente a traves de un puntero

    c.A cada uno de los elementos de la lista vinculada lo llamamos NODO

    d.Se unen al siguiente a traves de un puntero y al anterior a traves de otro puntero

7. Que es TDA? Ejemplifique

8. Dada la siguiente estructura y las líneas de codigo, responder:

   - ¿Que nodo queda como puntero al inicio de la lista?
   - Suponiendo tener una función que recorra y muestra los números: ¿Que orden de números tendrá como salida por pantalla?
   - Completar la línea 73 para mostrar todos los datos por pantalla.

```
50  typedef struct
51  {
52      int numero;
53      struct n * sig;
54  }n;
55
56  int main()
57  {
58      n * nuevo = (n*)malloc(sizeof(n));
59      nuevo->numero=1;
60      nuevo->sig=NULL;
61
62      n * nuevo2 = (n*)malloc(sizeof(n));
63      nuevo2->numero=2;
64      nuevo2->sig=NULL;
65
66      nuevo2->sig=nuevo;
67
68      n * nuevo3 = (n*)malloc(sizeof(n));
69      nuevo3->numero=3;
70      nuevo3->sig=NULL;
71
72      nuevo->sig=nuevo3;
73      recorrerYmostrar(.....................);
```

9. Reglas de la buena recursividad. Seleccione una o más de una:

   a. Al llegar a la Solución Trivial, produce un acercamiento a la Condición de Corte

   b. Al menos una llamada indirecta a la función

   c. Al llegar a la Solución Trivial, queda expresada la solución total.

   d. Al menos una condición trivial con su respectiva solución corte

   e. Al menos una llamada recursiva.

   f. Al menos una Condición de Corte con su respectiva Solución Trivial.

   g. En cada Llamada Recursiva, se produce un acercamiento a la Condición de Corte

10. Se desea desvincular el primer nodo de la lista y retornar el puntero al mismo. Complete las lineas de código que faltan.

```
350  ......... desvincularPrimerNodo(.......... lista)
351  {
352      nodo * aux=.............;
353      if(.............)
354      {
355          ........... =............ ->siguiente;
356
357          aux->siguiente=............ ;
358
359      }
360      return aux;
361  }
```

11. ¿Cuál es el problema de la recursividad?

    La recursividad invoca de manera repetida y puede ..... tanto en tiempo de procesador como en espacio de memoria. La lista vinculada es la estructura dinámica más simple, consiste de un conjunto de ...., creadas con malloc, y que no tienen nombre.

12. Seleccione LA o LAS opción/es que considera correcta/s para las características de la función: **malloc(..).** Seleccione una o más de una:

    a. Devuelve un nodo

    b. Prefiero no contestar

    c. Garantiza que la zona de memoria concedida no está ocupada por ninguna otra variable

    d. Devuelve un puntero inespecífico

    e. Devuelve la dirección de memoria de un entero

    f. Los valores almacenados en la dirección generada son nulos.

    g. Necesita conocer cuántos bytes se quieren reservar

    h. Si no es posible conceder el bloque de memoria (p.ej. no hay memoria suficiente), devuelve un puntero NULL

13. Explicar brevemente el objetivo de las siguientes funciones y qué valor se mostrará por consola.

```c
int funcA(int n);
int funcB(int n);

int funcA (int n)
{
    if (n==0)
       return 0;
    else
       return funcB(n-1);
}

int funcB (int n)
{
    if (n==0)
        return 1;
    else
        return funcA(n-1);
}
int main()
{
    printf("%d", funcA(4));
}
```

14. Una función recursiva puede invocar a otras funciones recursivas como parte de su solución. Seleccione una:

- Verdadero
- Falso

15. Una lista es un conjunto estático de nodos enlazados entre sí. Seleccione una:

- Verdadero
- Falso

16. Se tiene la siguiente estructura, que corresponde a una lista doblemente enlazada:

```
typedef struct {
    int valor;
    struct nodo2 * sig;
    struct nodo2 * ante;
} nodo2;
```

17. Al insertar un nodo, manteniendo el orden de sus datos de forma ascendente, ¿qué casos debo tener en cuenta?

18. Se tiene la siguiente info, se desea saber que hace la función, que devuelve y si tiene errores, deberá corregirlos.

```
typedef struct Nodo {
    int dato;
    struct Nodo* siguiente;
} Nodo;

void buscarUltimoNodo(Nodo** lista, Nodo** ultimo) {
    if (*lista == NULL) {
        ultimo = NULL;
        return;
    }

    Nodo* actual = *lista;
    while (actual->siguiente != NULL) {
        actual = actual->siguiente;
    }
    ultimo = actual;
}
```

19. Indique si lo siguiente función tiene errores y de ser así, justifique.

```
void asignarValor(int** ptrDoble) {
    int* valor = (int*)malloc(sizeof(int));
    *valor = 42;
    *ptrDoble = &valor;
}

int main() {
    int* puntero = NULL;
    asignarValor(&puntero);
    printf("El valor es: %d\n", *puntero);
    free(puntero);
    return 0;
}
```

20. Un puntero doble puede almacenar una dirección de memoria y/o un puntero simple. Indique V o F.

- Verdadero
- Falso