SEMESTRE MAYO- AGOSTO

Nombre: Jefferson David Yépez Morán		**NRC:** 14543
Docente: Ing. Nancy Betancourt		**Asignatura:** Estructura de datos
Fecha: 31/05/2024	**Deber # :**1	**Calificación:**

Documentación Actividad:

Ejercicio numero 1:

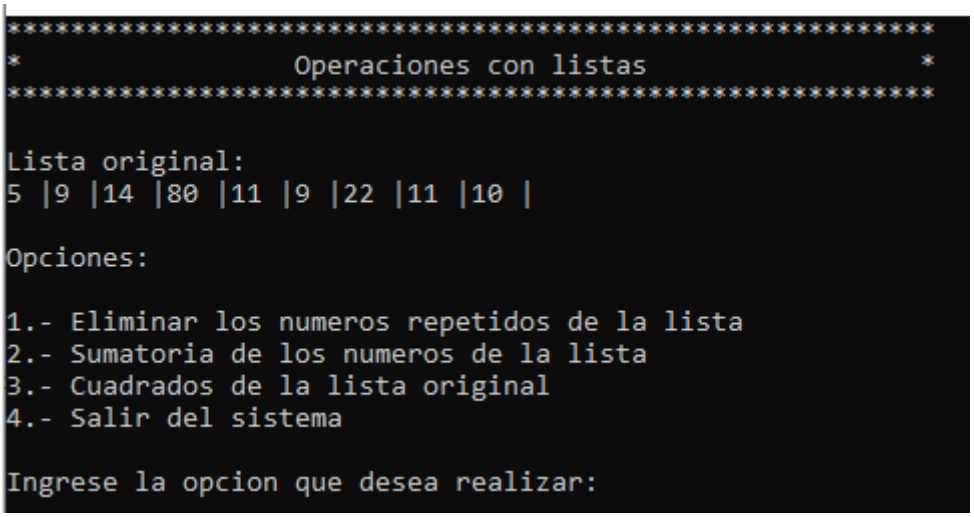
- Tras revisar los requisitos presentados en la primera actividad, se concluyó que la estructura más adecuada para llevarla a cabo es una lista doblemente enlazada. Esta estructura permite un acceso más eficiente a los elementos siguientes y previos, optimizando así el rendimiento de las operaciones necesarias.
- **Definiciones**:

1. La clase node se usó como una clase donde se definen los enlaces y el tipo de dato.
2. La clase lista se usó como una clase en el cual se indica el primer elemento de la lista, además de las funcionalidades a realizar.

```
class node {
public:
    int dato; // Variable para almacenar el dato del nodo.
    node *siguiente; // Puntero al siguiente nodo en la lista.
    node *previo; // Puntero al nodo previo en la lista.

    // Constructor de la clase node.
    node(int dato) {
        this->dato = dato; // Asigna el valor del dato al nodo.
        this->siguiente = NULL; // Inicializa el puntero al siguiente nodo como NULL.
        this->previo = NULL; // Inicializa el puntero al nodo previo como NULL.
    }
};
class lista {
public:
    node *head; // Puntero al primer nodo de la lista

    lista() {
        head = 0; // Inicializa la cabeza de la lista como null
    }

    // Declaraciones de métodos
    void insertarFinal(int dato);
    void imprimirLista();
    void eliminarNumeroRepetido();
    void calcularSumatoriaLista();
    lista listaCuadradosOriginales();
};
```

- Entre las funcionalidades se encuentran las siguiente:
  1. insertarFinal: Agrega un nodo al final de la lista con un dato tipo entero.
  2. imprimirLista: Imprime los elementos de la lista.
  3. eliminarNumeroRepetido: recorre la lista varias veces y busca elementos que sean iguales y los elimina dejando el primer elemento encontrado.
  4. calcularSumatoriaLista: recorre la lista recogiendo todos los datos de la misma y los suma en una variable.
  5. listaCuadradosOriginales: crea una nueva lista en la cual almacena los cuadrados de la lista original.

- **Intefaz de Usuario:**



Ejercicio numero 2:

- Tras revisar los requisitos presentados en la segunda actividad, se concluyó que la estructura más adecuada para llevarla a cabo es una lista circular doblemente enlazada, ya que se requiere simular una ruleta y para ello se debe recorrer la lista en ambos sentidos.
- **Definiciones:**
  1. La clase node se usó como una clase donde se definen los enlaces y el tipo de dato.
  2. La clase lista se usó como una clase en el cual se indica el primer elemento de la lista, además de las funcionalidades a realizar.

SEMESTRE MAYO- AGOSTO

```
class node {
public:
    string premio; // Variable para almacenar el premio del nodo.
    node *siguiente; // Puntero al siguiente nodo en la lista.
    node *previo; // Puntero al nodo previo en la lista.

    // Constructor de la clase node.
    node(string premio) {
        this->premio = premio; // Asigna el valor del premio al nodo.
        this->siguiente = NULL; // Inicializa el puntero al siguiente nodo como NULL.
        this->previo = NULL; // Inicializa el puntero al nodo previo como NULL.
    }
};
class lista {
public:
    node *head; // Puntero al primer nodo de la lista.

    lista() {
        head = 0; // Inicializa el puntero 'head' como NULL.
    }

    void insertarPremios(string premio); // Función para insertar un premio en la list
    void imprimirLista(); // Función para imprimir todos los premios en la lista.
    int generarNumeroAleatorio(); // Función para generar un número aleatorio entre 2
    void girarRuleta(); // Función para "girar" la ruleta y seleccionar un premio ales
};
```

- Entre las funcionalidades se encuentran las siguiente:
  1. insertarPremios: Agrega un nodo al final de la lista con un premio de tipo string.
  2. imprimirLista: Imprime los elementos de la lista.
  3. generarNumeroAleatorio: genera un numero aleatorio del 2 al 7.
  4. girarRuleta: realiza la simulacion de una ruleta, en este caso retrocede un movimiento y avanza aleatoriamente de 2 a 7 elementos siguientes y devuelve el premio.

- **Intefaz de Usuario:**

```
**********************************************************
*              Bienvenido a la Ruleta                    *
**********************************************************

Premios de la ruleta:
Premio 1 es: Balon
Premio 2 es: rompecabezas
Premio 3 es: mouse
Premio 4 es: teclado

1.- Girar la ruleta
2.- Salir de la ruleta

Ingrese la opcion que desea realizar:
```