

UNIVERSIDAD NACIONAL DE SAN AGUSTIN

CIENCIA DE LA COMPUTACION II

LABORATORIO 10

Alumno: Alexandro Ysaác Delgado Justo

CUI: 20173455

Profesor: Enzo Velázquez Lobatón

Repositorio: <https://github.com/adelgadoj/CCII-LABORATORIO10>

1. Defina una lista enlazada que permita insertar elementos al final de todos los elementos que ya se hayan ingresado. Por el momento no es necesario preservar un orden, simplemente los elementos nuevos deben de ingresar como el último elemento.

Nodo.h

```
#include <iostream>
class Nodo
{
private:
    int dato;
    Nodo *next;

public:
    Nodo(int);
    int getDato();
    void setDato(int);
    Nodo *getNext();
    void setNext(Nodo *);
};
Nodo::Nodo(int _dato)
{
    dato = _dato;
    next = nullptr;
}
int Nodo::getDato()
{
    return dato;
}

void Nodo::setDato(int _dato)
{
    dato = _dato;
}

Nodo *Nodo::getNext()
{
    return next;
}
void Nodo::setNext(Nodo *_next)
{

```

```
    next = _next;
}
```

ListaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void print();
};

ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}

ListaEnlazada::~ListaEnlazada()
{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}

void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {
            aux = aux->getNext();
        }
        aux->setNext(temp);
        size++;
    }
}

void ListaEnlazada::print()
{
    Nodo *aux = head;
```

```

while (aux)
{
    cout << aux->getDato() << " ";
    aux = aux->getNext();
}
cout<<endl;
}

```

Main.cpp

```

#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu(){
    while (true)
    {
        int r;
        int d;
        cout<<"\tMENU"<<endl;
        cout<<"Elija Opcion..."<<endl;
        cout<<"1) Ingresar Al Final"<<endl;
        cout<<"2) Mostrar Datos"<< endl;
        cout<<"0) Salir"<<endl;
        cout<<"R: "<<endl;
        cin>>r;
        switch (r)
        {
            case 0:
                exit(1);
            case 1:
                cout<<"Ingrese Nuevo Dato: "<<endl;
                cin>>d;
                l.insertarFinal(d);
                break;
            case 2:
                l.print();
                system("pause");
                break;
            default:
                cout<<"Opcion No Valida"<<endl;
                system("pause");
                break;
        }
    }
}

int main()
{
    menu();
}

```

```
    return 0;  
}
```

Captura Programa Ejercicio 1:

```
C:\Users\aldej\Desktop\PROGRAMAS\CC- II\LAB10\Ejercicio_1>1.exe  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
1  
Ingrese Nuevo Dato:  
1  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
1  
Ingrese Nuevo Dato:  
2  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
1  
Ingrese Nuevo Dato:  
3  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
1  
Ingrese Nuevo Dato:  
4  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
1  
Ingrese Nuevo Dato:  
5  
    MENU  
Elija Opcion...  
1) Ingresar Al Final  
2) Mostrar Datos  
0) Salir  
R:  
2  
1 2 3 4 5
```

2. Con la implementación de la lista enlazada anterior, desarrolle una función que permita ingresar los elementos al inicio de todos los demás elementos. Tendrá que modificar el comportamiento del puntero que tiene referencia al primer elemento para que sea redireccionado al nuevo elemento por ingresar.

Solo se modifiko el archivo main.cpp y listaEnlazada.h

listaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void insertarPrincipio(int);
    void print();
};

ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}

ListaEnlazada::~ListaEnlazada()
{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}

void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {

```

```

        aux = aux->getNext();
    }
    aux->setNext(temp);
    size++;
}
}
//Funcion Nueva Para insertar elementos al principio de la lista
void ListaEnlazada::insertarPrincipio(int _dato)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if (head == nullptr)
    {
        head = temp;
    }
    else
    {
        temp->setNext(aux);
        head = temp;
    }
}
void ListaEnlazada::print()
{
    Nodo *aux = head;
    while (aux)
    {
        cout << aux->getDato() << " ";
        aux = aux->getNext();
    }
    cout<<endl;
}

```

main.cpp

```

#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu()
{
    while (true)
    {
        int r;
        int d;
        cout << "\tMENU" << endl;
        cout << "Elija Opcion..." << endl;
        cout << "1) Ingresar Al Final" << endl;
        cout << "2) Ingresar Al Principio" << endl;
        cout << "3) Mostrar Datos" << endl;
        cout << "0) Salir" << endl;
        cout << "R: " << endl;
    }
}

```

```

cin >> r;

switch (r)
{
case 0:
    exit(1);
case 1:
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarFinal(d);
    break;
case 2:
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarPrincipio(d);
    break;
case 3:
    l.print();
    system("pause");
    break;
default:
    cout << "Opcion No Valida" << endl;
    system("pause");
    break;
}
}

int main()
{
    menu();
    return 0;
}

```

Captura pantalla progra

```
C:\Users\aldej\Desktop\PROGRAMAS\CC- II\LAB10\Ejercicio_2>2.exe
      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Mostrar Datos
0) Salir
R:
1
Ingrese Nuevo Dato:
1
      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Mostrar Datos
0) Salir
R:
1
Ingrese Nuevo Dato:
2
      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Mostrar Datos
0) Salir
R:
1
Ingrese Nuevo Dato:
3
      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Mostrar Datos
0) Salir
R:
2
Ingrese Nuevo Dato:
4
      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Mostrar Datos
0) Salir
R:
3
4 1 2 3
Presione una tecla para continuar . . .
```


3. Desarrolle una función que permita ingresar elementos en el medio de dos elementos de la lista enlazada, como se muestra en la siguiente imagen. Solicite que se ingrese una posición válida dentro de la lista y permita que el valor ingresado se pueda anexar en esa posición.

listaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void insertarPrincipio(int);
    void insertarEnMedio(int, int);
    int getSize();
    void print();
};
ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}
ListaEnlazada::~ListaEnlazada()
{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}
void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {
```

```

        aux = aux->getNext();
    }
    aux->setNext(temp);

}
size++;
}
void ListaEnlazada::insertarPrincipio(int _dato)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if (head == nullptr)
    {
        head = temp;
    }
    else
    {
        temp->setNext(aux);
        head = temp;
    }
    size++;
}
//Funcion Nueva Para insertar elementos en medio de la lista
void ListaEnlazada::insertarEnMedio(int _dato, int _pos)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr)
    {
        head = temp;
    }
    else{
        for(int i = 1 ; i <= _pos ; i++){
            if(i == _pos){
                temp->setNext(aux->getNext());
                aux->setNext(temp);
            }
            aux = aux->getNext();
        }
    }
    size++;
}

void ListaEnlazada::print()
{
    Nodo *aux = head;
    while (aux)
    {
        cout << aux->getDato() << " ";
        aux = aux->getNext();
    }
    cout<<endl;
}

```

```

}
int ListaEnlazada::getSize()
{
    return size;
}

```

Main.cpp

```

#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu()
{
    while (true)
    {
        int r;
        int d;
        cout << "\tMENU" << endl;
        cout << "Elija Opcion..." << endl;
        cout << "1) Ingresar Al Final" << endl;
        cout << "2) Ingresar Al Principio" << endl;
        cout << "3) Ingresar dato en posicion X " << endl;
        cout << "4) Mostrar Datos" << endl;
        cout << "0) Salir" << endl;
        cout << "R: " << endl;
        cin >> r;
        switch (r)
        {
            case 0:
                exit(1);
            case 1:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarFinal(d);
                break;
            case 2:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarPrincipio(d);
                break;
            case 3:
                int pos;
                cout << "Ingrese Posicion Valida: " << endl;
                cin >> pos;
                if(pos > l.getSize()-1 || pos < 0){
                    cout << "!!Posicion Invalida!! " << endl;
                }
                else if(pos == 0){
                    cout << "Ingrese Nuevo Dato: " << endl;
                    cin >> d;

```

```

        l.insertarPrincipio(d);
        break;
    }
    else{
        cout << "Ingrese Nuevo Dato: " << endl;
        cin >> d;
        l.insertarEnMedio(d,pos);
    }
    break;
case 4:
    l.print();
    system("pause");
    break;
default:
    cout << "Opcion No Valida" << endl;
    system("pause");
    break;
}
}
}

int main()
{
    menu();
    return 0;
}

```

Captura del programa:

```

      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Mostrar Datos
0) Salir
R:
4
1 2 3 4
Presione una tecla para continuar . . .

      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Mostrar Datos
0) Salir
R:
3
Ingrese Posicion Valida:
2
Ingrese Nuevo Dato:
10

      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Mostrar Datos
0) Salir
R:
4
1 2 10 3 4
Presione una tecla para continuar . . .
```

4. Elabore una función que permita eliminar el último elemento de una lista enlazada.

(Evite copiar los elementos en una nueva lista para completar la eliminación del elemento)

listaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void insertarPrincipio(int);
    void insertarEnMedio(int, int);
    void removeUltimo();
    int getSize();
    void print();
};

ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}

ListaEnlazada::~ListaEnlazada()
{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}

void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {
            aux = aux->getNext();
        }
    }
}
```

```

        aux->setNext(temp);

    }
    size++;
}
void ListaEnlazada::insertarPrincipio(int _dato)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if (head == nullptr)
    {
        head = temp;
    }
    else
    {
        temp->setNext(aux);
        head = temp;
    }
    size++;
}
void ListaEnlazada::insertarEnMedio(int _dato, int _pos)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr)
    {
        head = temp;
    }
    else{
        for(int i = 1 ; i <= _pos ; i++){
            if(i == _pos){
                temp->setNext(aux->getNext());
                aux->setNext(temp);
            }
            aux = aux->getNext();
        }
    }
    size++;
}
void ListaEnlazada::removeUltimo()
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout<<"##Lista vacia##"<<endl;
        system("pause");
        return;
    }
    else
    {
        Nodo *aux = head;
        int cont = 0;

```

```

        while (cont != size - 2)
        {
            aux = aux->getNext();
            cont++;
        }
        delete aux->getNext();
        aux->setNext(nullptr);
        size--;
    }
}

void ListaEnlazada::print()
{
    Nodo *aux = head;
    while (aux)
    {
        cout << aux->getDato() << " ";
        aux = aux->getNext();
    }
    cout<<endl;
}

int ListaEnlazada::getSize()
{
    return size;
}

```

Main.cpp

```

#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu()
{
    while (true)
    {
        int r;
        int d;
        cout << "\tMENU" << endl;
        cout << "Elija Opcion..." << endl;
        cout << "1) Ingresar Al Final" << endl;
        cout << "2) Ingresar Al Principio" << endl;
        cout << "3) Ingresar dato en posicion X " << endl;
        cout << "4) Eliminar Ultimo Elemento" << endl;
        cout << "5) Mostrar Datos" << endl;
        cout << "0) Salir" << endl;
        cout << "R: " << endl;
        cin >> r;
        switch (r){
        case 0:
            exit(1);

```



```

case 1:
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarFinal(d);
    break;
case 2:
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarPrincipio(d);
    break;
case 3:
    int pos;
    cout << "Ingrese Posicion Valida: " << endl;
    cin >> pos;
    if(pos > l.getSize()-1 || pos < 0){
        cout << "!!Posicion Invalida!! " << endl;
    }
    else if(pos == 0){
        cout << "Ingrese Nuevo Dato: " << endl;
        cin >> d;
        l.insertarPrincipio(d);
        break;
    }
    else{
        cout << "Ingrese Nuevo Dato: " << endl;
        cin >> d;
        l.insertarEnMedio(d,pos);
    }
    break;
case 4:
    l.print();
    l.removeUltimo();
    cout << "Se Elimino ultimo elemento" << endl;
    l.print();
    break;
case 5:
    l.print();
    system("pause");
    break;

default:
    cout << "Opcion No Valida" << endl;
    system("pause");
    break;
}
}
}

int main(){
    menu();
    return 0;
}

```

Captura del programa:

```
MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Eliminar Ultimo Elemento
5) Mostrar Datos
0) Salir
R:
4
1 2 3 4 5
Se Elimino ultimo elemento
1 2 3 4
```

5. Desarrolle una función que permita eliminar el primer elemento de una lista sin perder referencia de los demás elementos que ya se encuentran almacenados en la estructura .(Evite copiar los elementos en una nueva lista para completar la eliminación de los elementos)

listaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void insertarPrincipio(int);
    void insertarEnMedio(int, int);
    void removeUltimo();
    void removePrimero();
    int getSize();
    void print();
};
ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}
ListaEnlazada::~ListaEnlazada()
```

```

{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}

void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {
            aux = aux->getNext();
        }
        aux->setNext(temp);
    }
    size++;
}

void ListaEnlazada::insertarPrincipio(int _dato)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if (head == nullptr)
    {
        head = temp;
    }
    else
    {
        temp->setNext(aux);
        head = temp;
    }
    size++;
}

void ListaEnlazada::insertarEnMedio(int _dato, int _pos)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr)
    {
        head = temp;
    }
    else{
        for(int i = 1 ; i <= _pos ; i++){
            if(i == _pos){
                temp->setNext(aux->getNext());
            }
        }
    }
}

```

```

        aux->setNext(temp);
    }
    aux = aux->getNext();
}
size++;
}
void ListaEnlazada::removeUltimo()
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout<<"##Lista vacia##"<<endl;
        system("pause");
        return;
    }
    else
    {
        Nodo *aux = head;
        int cont = 0;
        while (cont != size - 2)
        {
            aux = aux->getNext();
            cont++;
        }
        delete aux->getNext();
        aux->setNext(nullptr);
        size--;
    }
}
void ListaEnlazada::removePrimero()
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout << "##Lista vacia##" << endl;
        system("pause");
        return;
    }
    else
    {
        Nodo *aux = head;
        head = aux->getNext();
        delete aux;
        size--;
    }
}

void ListaEnlazada::print()
{
    Nodo *aux = head;
    while (aux)

```

```

    {
        cout << aux->getDato() << " ";
        aux = aux->getNext();
    }
    cout<<endl;
}
int ListaEnlazada::getSize()
{
    return size;
}

```

Main.cpp

```

#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu()
{
    while (true)
    {
        int r;
        int d;
        cout << "\tMENU" << endl;
        cout << "Elija Opcion..." << endl;
        cout << "1) Ingresar Al Final" << endl;
        cout << "2) Ingresar Al Principio" << endl;
        cout << "3) Ingresar dato en posicion X " << endl;
        cout << "4) Eliminar Ultimo Elemento" << endl;
        cout << "5) Eliminar Primer Elemento" << endl;
        cout << "6) Mostrar Datos" << endl;
        cout << "0) Salir" << endl;
        cout << "R: " << endl;
        cin >> r;
        switch (r)
        {
            case 0:
                exit(1);
            case 1:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarFinal(d);
                break;
            case 2:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarPrincipio(d);
                break;
            case 3:
                int pos;
                cout << "Ingrese Posicion Valida: " << endl;
                cin >> pos;

```

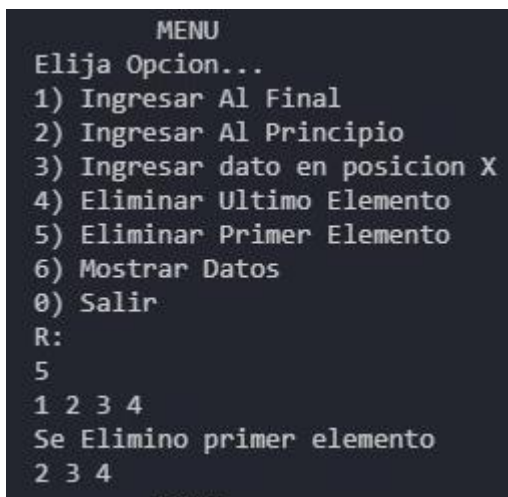
```

if(pos > l.getSize()-1 || pos < 0){
    cout << "!!Posicion Invalida!! " << endl;
}
else if(pos == 0){
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarPrincipio(d);
    break;
}
else{
    cout << "Ingrese Nuevo Dato: " << endl;
    cin >> d;
    l.insertarEnMedio(d,pos);
}
break;
case 4:
    l.print();
    l.removeUltimo();
    cout << "Se Elimino ultimo elemento" << endl;
    l.print();
    break;
case 5:
    l.print();
    l.removePrimero();
    cout << "Se Elimino primer elemento" << endl;
    l.print();
    break;
case 6:
    l.print();
    system("pause");
    break;

default:
    cout << "Opcion No Valida" << endl;
    system("pause");
    break;
}

```

Captura de Pantalla:



```

      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Eliminar Ultimo Elemento
5) Eliminar Primer Elemento
6) Mostrar Datos
0) Salir
R:
5
1 2 3 4
Se Elimino primer elemento
2 3 4
      MENU

```

6. Dado una posición válida dentro de la lista, permita al usuario eliminar un elemento de cualquier posición sin perder referencia de los demás elementos.

listaEnlazada.h

```
#include <iostream>
#include "nodo.h"
using namespace std;
class ListaEnlazada
{
private:
    Nodo *head;
    int size;

public:
    ListaEnlazada();
    ~ListaEnlazada();
    void insertarFinal(int);
    void insertarPrincipio(int);
    void insertarEnMedio(int, int);
    void removeUltimo();
    void removePrimero();
    void remove(int);
    int getSize();
    void print();
};

ListaEnlazada::ListaEnlazada()
{
    head = nullptr;
    size = 0;
}

ListaEnlazada::~ListaEnlazada()
{
    Nodo *aux = head;
    while (head != nullptr)
    {
        aux = head->getNext();
        delete head;
        head = aux;
    }
}

void ListaEnlazada::insertarFinal(int _dato){
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr){
        head = temp;
    }
    else{
        while (aux->getNext() != nullptr)
        {
            aux = aux->getNext();
        }
    }
}
```

```

        aux->setNext(temp);

    }
    size++;
}
void ListaEnlazada::insertarPrincipio(int _dato)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if (head == nullptr)
    {
        head = temp;
    }
    else
    {
        temp->setNext(aux);
        head = temp;
    }
    size++;
}
void ListaEnlazada::insertarEnMedio(int _dato, int _pos)
{
    Nodo *temp = new Nodo(_dato);
    Nodo *aux = head;
    if(head == nullptr)
    {
        head = temp;
    }
    else{
        for(int i = 1 ; i <= _pos ; i++){
            if(i == _pos){
                temp->setNext(aux->getNext());
                aux->setNext(temp);
            }
            aux = aux->getNext();
        }
    }
    size++;
}
void ListaEnlazada::removeUltimo()
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout<<"##Lista vacia##"<<endl;
        system("pause");
        return;
    }
    else
    {
        Nodo *aux = head;
        int cont = 0;

```



```

        while (cont != size - 2)
        {
            aux = aux->getNext();
            cont++;
        }
        delete aux->getNext();
        aux->setNext(nullptr);
        size--;
    }
}

void ListaEnlazada::removePrimero()
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout << "##Lista vacia##" << endl;
        system("pause");
        return;
    }
    else
    {
        Nodo *aux = head;
        head = aux->getNext();
        delete aux;
        size--;
    }
}

void ListaEnlazada::remove(int pos) //FUNCION PARA BORRRAR CUALQUIER ELEMENTO
{
    Nodo *aux = head;
    if (head == nullptr)
    {
        cout<<"Lista Vacia"<<endl;
        system("pause");
        return;
    }
    else if (pos > size)
    {
        cout<<"Ingrese Posicion Valida"<<endl;
        system("pause");
        return;
    }
    else
    {
        if (pos == size)
        {
            Nodo *aux = head;
            int cont = 0;
            while (cont != pos - 2)
            {
                aux = aux->getNext();
                cont++;
            }

```

```

        }
        delete aux->getNext();
        aux->setNext(nullptr);
    }
    else if (pos == 1)
    {
        Nodo *aux = head;
        head = aux->getNext();
        delete aux;
    }
    else
    {
        Nodo *aux = head;
        int cont = 0;
        while (pos - 2 != cont)
        {
            aux = aux->getNext();
            cont++;
        }
        Nodo *aux2 = aux->getNext();
        aux->setNext(aux2->getNext());
        delete aux2;
    }
    size--;
}

void ListaEnlazada::print()
{
    Nodo *aux = head;
    while (aux)
    {
        cout << aux->getDato() << " ";
        aux = aux->getNext();
    }
    cout<<endl;
}

int ListaEnlazada::getSize()
{
    return size;
}

```

Main.cpp

```
#include <iostream>
#include "ListaEnlazada.h"
using namespace std;
ListaEnlazada l;
void menu()
{
    while (true)
    {
        int r;
        int d;
        cout << "\tMENU" << endl;
        cout << "Elija Opcion..." << endl;
        cout << "1) Ingresar Al Final" << endl;
        cout << "2) Ingresar Al Principio" << endl;
        cout << "3) Ingresar dato en posicion X " << endl;
        cout << "4) Eliminar Ultimo Elemento" << endl;
        cout << "5) Eliminar Primer Elemento" << endl;
        cout << "6) Eliminar Elemento en posicion X" << endl;
        cout << "7) Mostrar Datos" << endl;
        cout << "0) Salir" << endl;
        cout << "R: " << endl;
        cin >> r;
        switch (r)
        {
            case 0:
                exit(1);
            case 1:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarFinal(d);
                break;
            case 2:
                cout << "Ingrese Nuevo Dato: " << endl;
                cin >> d;
                l.insertarPrincipio(d);
                break;
            case 3:
                int pos;
                cout << "Ingrese Posicion Valida: " << endl;
                cin >> pos;
                if(pos > l.getSize()-1 || pos < 0){
                    cout << "!!Posicion Invalida!! " << endl;
                }
                else if(pos == 0){
                    cout << "Ingrese Nuevo Dato: " << endl;
                    cin >> d;
                    l.insertarPrincipio(d);
                    break;
                }
            else{
```

```

        cout << "Ingrese Nuevo Dato: " << endl;
        cin >> d;
        l.insertarEnMedio(d,pos);
    }
    break;
case 4:
    l.print();
    l.removeUltimo();
    cout << "Se Elimino ultimo elemento" << endl;
    l.print();
    break;
case 5:
    l.print();
    l.removePrimero();
    cout << "Se Elimino primer elemento" << endl;
    l.print();
    break;
case 6:
    int posi;
    l.print();
    cout << "Indique la posicion del elemento a Eliminar:"<<endl;
    cin >> posi;
    l.remove(posi+1);
    cout << "Se Elimino elemento" << endl;
    l.print();
    break;
case 7:
    l.print();
    system("pause");
    break;

default:
    cout << "Opcion No Valida" << endl;
    system("pause");
    break;
}
}
}

int main()
{
    menu();
    return 0;
}

```

Captura del Programa:

```

      MENU
Elija Opcion...
1) Ingresar Al Final
2) Ingresar Al Principio
3) Ingresar dato en posicion X
4) Eliminar Ultimo Elemento
5) Eliminar Primer Elemento
6) Eliminar Elemento en posicion X
7) Mostrar Datos
0) Salir
R:
6
1 2 3 4 5
Indique la posicion del elemento a Eliminar:
2
Se Elimino elemento
1 2 4 5
```