Grupo - 2

LISTAS DOBLEMENTE ENLAZADAS CIRCULARES

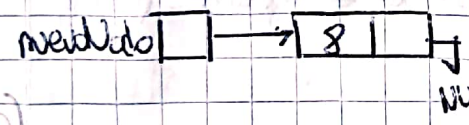
```
class Lista {

    private:
        Nodo * primero;
        Nodo * ultimo;
        Nodo * nuevoNodo;
    public:
        Lista() = default;
        bool estaVacia();
        void agregarPorCabeza(int);
        void agregarPorCola(int);
        void agregarEntre(int, int);
        void eliminarPorCabeza();
        void eliminarPorCola();
        void eliminarEntre(int);

};
```

```
class Nodo {

    private:
        int dato;
        Nodo * siguiente;
        Nodo * anterior;
    public:

        Nodo(int);

};
```

Alemán José
Han Martín
Sarmiento Alejandro.
Bolaños Mateo.

1er Método. Agregar Por Cabeza.

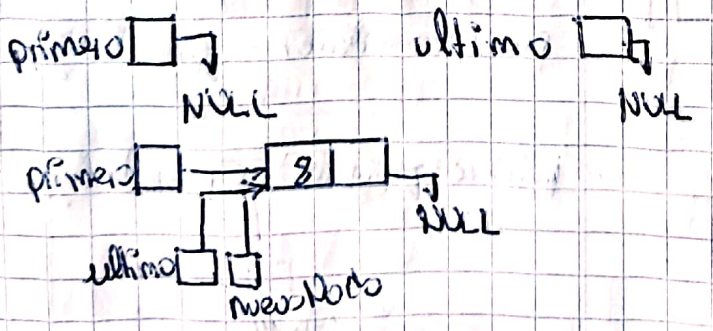
a) Se crea el nuevo Nodo con identificador de valor 8.




```
nuevoNodo = new Nodo(8);
nuevoNodo->siguiente = NULL;
```

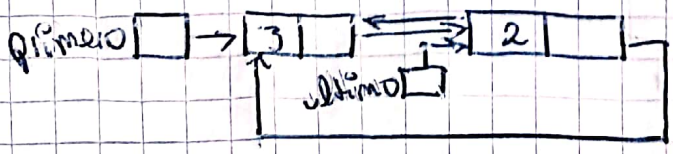
b)

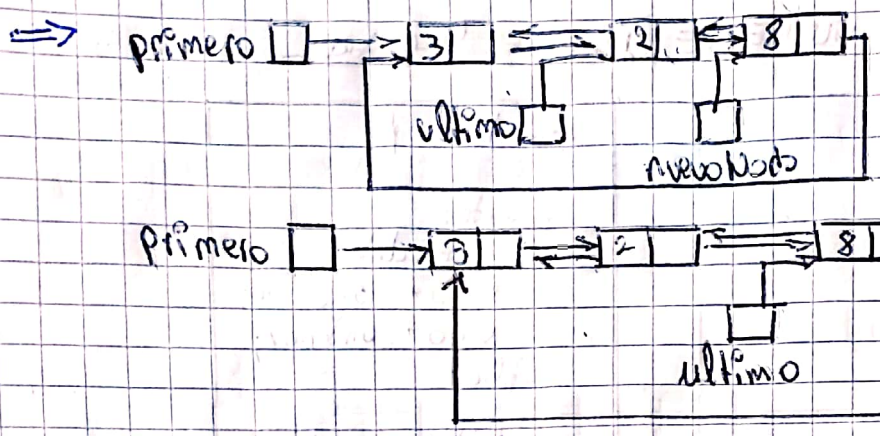
b.1) Suponiendo que la lista está vacía:




```
primero = nuevoNodo;
ultimo = nuevoNodo;
```

b.2) Suponiendo que la lista ya contenía elementos:

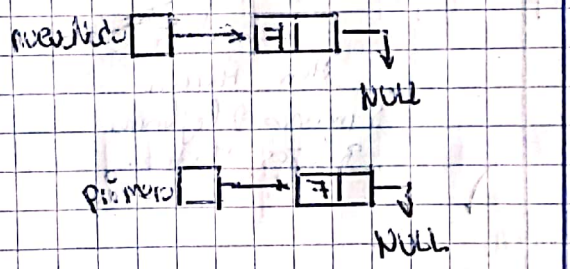




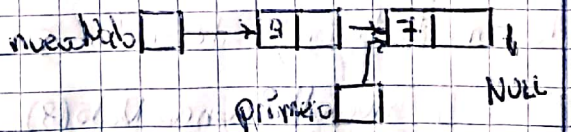
ultimo->siguiente = nuevoNodo;
nuevoNodo->siguiente = primero;
nuevoNodo->anterior = ultimo;

ultimo = nuevoNodo;

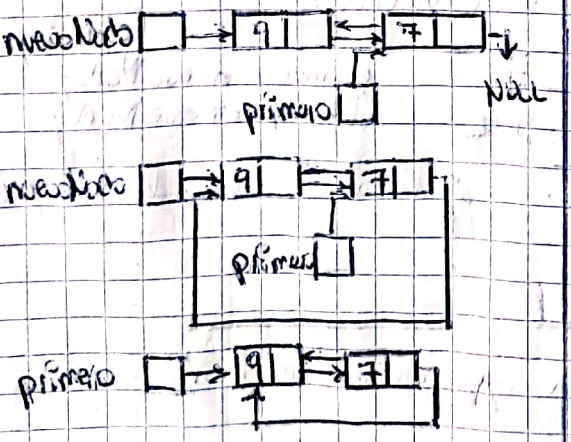
2do Método. Agregar por Cola.



nuevoNodo = new Nodo(7);
nuevoNodo->siguiente = primero;



nuevoNodo = new Nodo(9);
nuevoNodo->siguiente = primero;

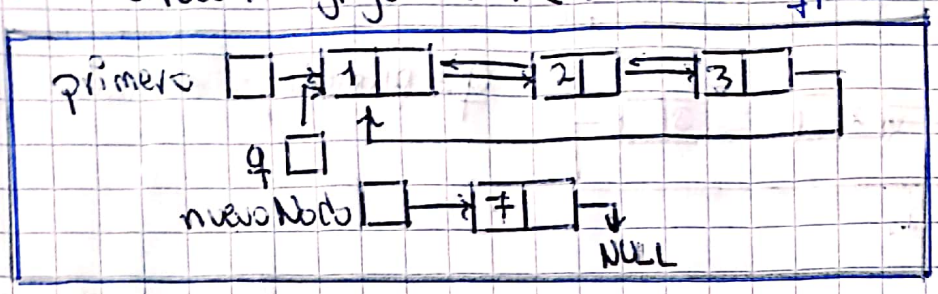


primero->anterior = nuevoNodo;

primero->siguiente = nuevoNodo;

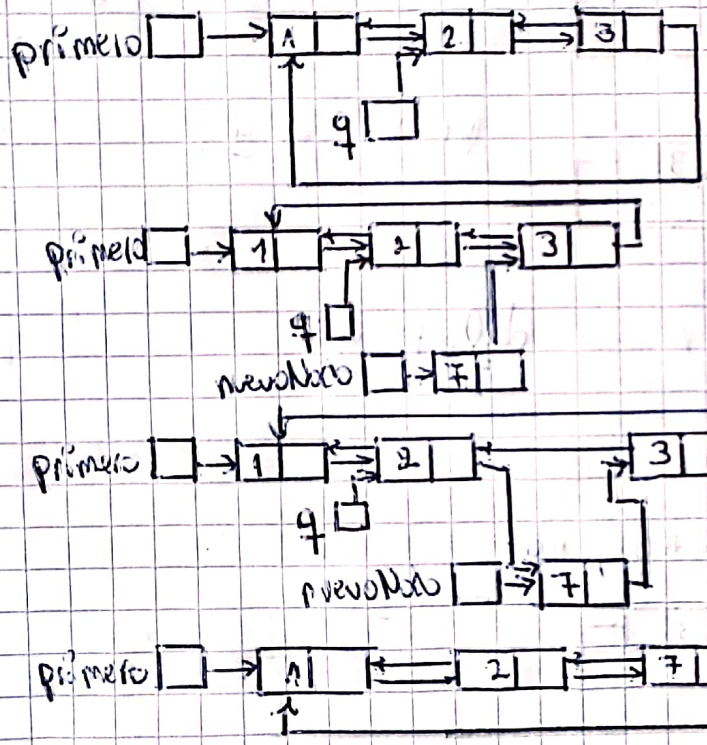
primero = nuevoNodo;

3er Método. Agregar antes. Nodo* q;



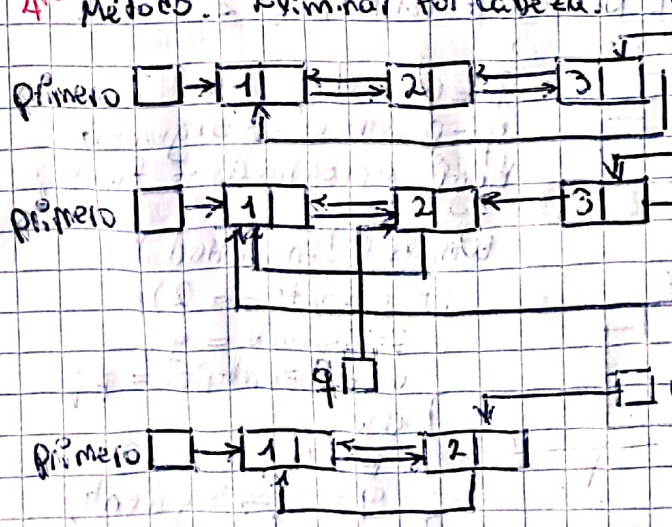
VALOR DEL IDENTIFICADOR DEL NODO A BUSCAR:

n = 2



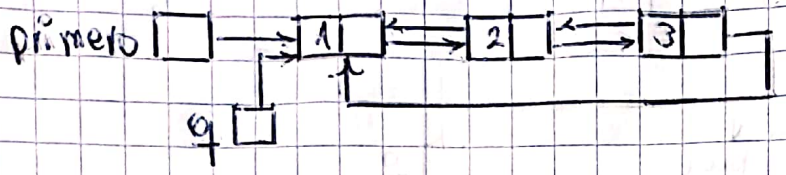

```
q = primero;
bool encontrado = false;
while(!encontrado){
  if(q->dato == n){
    nuevoNodo->siguiente = q->siguiente;
    q->siguiente = nuevoNodo;
    nuevoNodo->siguiente->anterior = nuevoNodo;
  } else
    q = q->siguiente;
}
```

4to Método. Eliminar por Cabeza. Nodo* q;

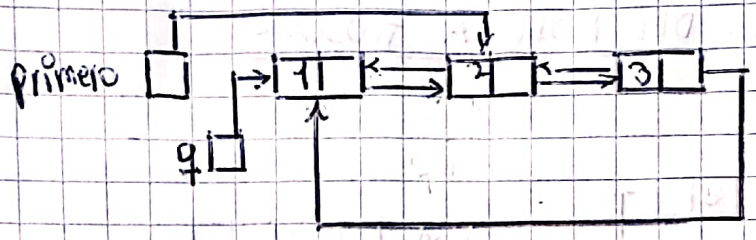



```
q = ultimo->anterior;
q->siguiente = primero;
delete ultimo;
ultimo = q;
```

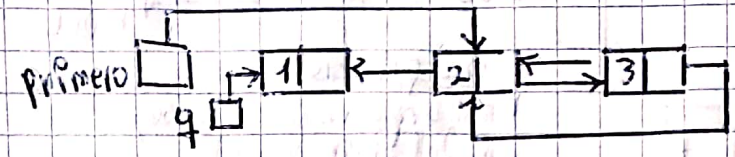
5to. Método. Eliminar Por Cola. Nodo *q;

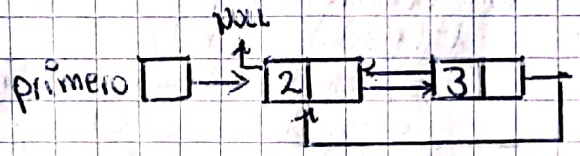


q = primero;



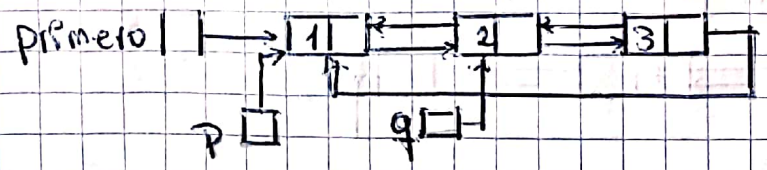
primero = primero → siguiente;



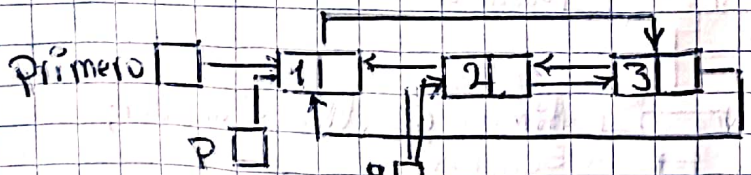


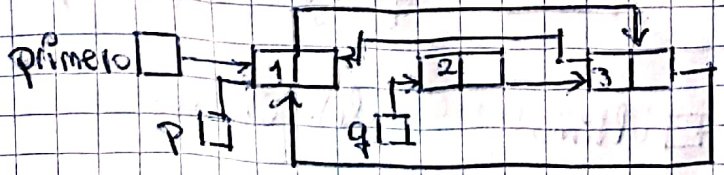
delete q;

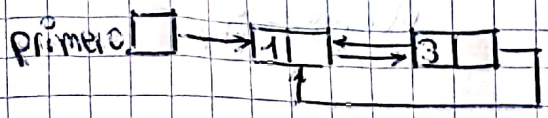
6to Método. Eliminar Entre. Nodo *p, *q;



VALOR DEL IDENTIFICADOR DEL NODO A ELIMINAR : 2








```
p = primero;
q = primero → siguiente;
bool encontrado = false;
while(!encontrado){
  if(q→dato == 2){
    p→siguiente = q→siguiente;
    ultimo→anterior = p;
  }else{
    p = q;
    q = q→siguiente;
  }
}
delete q;
```