UNIVERSIDAD AUTÓNOMA GABRIEL RENE MORENO
FACULTAD DE INGENIERÍA Y CIENCIAS DE LA COMPUTACIÓN Y TELECOMUNICACIONES

ESTRUCTURAS DE DATOS 2

CONTENIDO: *// Breve Descripción del Contenido*

LAB-6. ELIMINAR NODOS, LISTAS DOBLES.

PORCENTAJE TERMINADO : 100%.

GRUPO: 14

Nombre	Registro
Cristian Gabriel Gedalge Cayhuara	219022062

Fecha de Presentación: Jueves ,11 de abril de 2024

COMENTARIO:
Me parecieron desafiantes e interesante los ejercicios propuestos en este laboratorio, pude entender más sobre la importancia de los punteros que fueron fundamentales para resolver los ejercicios.

1. L1.eliminarPrim() : Método que elimina el elemento de la primer posición.

```
    public void eliminarPrim(){
        if (vacia()) {
            return ; }
            if (prim == ult ) {
                prim = ult = null ;
            }else{
                prim.prox.ant = null ;
                prim= prim.prox ;

            }
            cantElem-- ;

    }
```

2. L1.eliminarUlt() : Método que elimina el último elemento de la lista L1.

```
public void eliminarUlt(){
        if (vacia()) {
            return ;
            }
        if (prim==ult) {
            prim= ult = null ;

        }else{
            ult.ant.prox = null ;
            ult = ult.ant  ;

        }
        cantElem--;

    }
```

3. L1.eliminarNodo(ap, p) : Método que elimina el nodo p, y devuelve el nodo el nodos siguiente a ap. El nodo p, puede estar al principio, final o al centro de la lista.

```
public Nodo eliminarNodo(Nodo ap,Nodo p)
    {
        if(p==null)return null;
        if(ap==null)
        {
            eliminarPrim();
            return prim;
```

```
        }
        if(p.prox==null)
        {
            eliminarUlt();
            return null;
        }
        ap.prox=p.prox;
        p.prox.ant=ap;
        cantElem--;
        return ap.prox;
    }
```

4. L1.eliminarTodo(x) : Método que elimina todos los elementos x de la lista L1.

```
   public void  eliminarTodo(int x)
    {
        Nodo p=this.prim,ap=null;
        while (p!=null)
        {
            if(p.elem==x)
            {
                ap.prox=eliminarNodo(ap,p);
                p=p.prox;
            }else{
                ap=p;
                p=p.prox;
            }
        }
    }
```

5. L1.eliminarPrim(n) : Método que eliminar los primeros n-elementos de la lista L1.

```
public void eliminarPrim(int n)
    {
        if(vacia())
        {
            return ;
        }
        while(prim!=null &&n>0)
        {
            eliminarPrim();
            n--;
        }
    }
```

6. L1.eliminarUlt(n) : Método que elimina los n-últimos elementos de la lista L1.

```
public void eliminarUlt(int n)
    {
        if(vacia())
        {
            return ;
        }
        while(prim!=null &&n>0)
        {
            eliminarUlt();
            n--;
        }
    }
```

7. L1.eliminarIesimo(i) : Método que elimina el i-ésimo elemento de la lista L1.

```
public void eliminarIesimo(int i) {
        int k = 0;
        Nodo p = prim, ap = null;
        while (k < i && p != null) {
            ap = p;
            p = p.prox;
            k = k + 1;
        }
        eliminarNodo(ap, p);
    }
```