RESOLUCIÓN
PRACTICA NRO.
10

Clase Main

```
public static void main(String[] args) {
      // TODO code application logic here
      Lista nuevo=new Lista();
      nuevo.insertar(5);
      nuevo.insertar(3);
      nuevo.insertar(8);
      nuevo.insertar(1);
      nuevo.insertar(4);

      nuevo.mostrarID();

      nuevo.mostrarDI();
   }
```

Clase Lista

```
public class Lista {
   Nodo Inicio;
   Nodo ultimo;
   int tam;
   public Lista(){
      tam=0;
   }

   public void insertar(int dato){
      Nodo aux=new Nodo(dato);
      if(tam==0){
         Inicio=aux;
      }else{
         ultimo.posterior=aux;
         aux.anterior=ultimo;
      }
      ultimo=aux;
      tam++;
   }
   public void mostrarID(){
      Nodo aux=Inicio;
      while(aux!=null){
         System.out.println(aux.info);
         aux=aux.posterior;
      }
   }
   public void mostrarDI(){
      Nodo aux=ultimo;
      while(aux!=null){
         System.out.println(aux.info);
```

```
        aux=aux.anterior;
      }
   }
}
```

Clase Nodo

```
public class Nodo {
   int info;
   Nodo anterior;
   Nodo posterior;

   public Nodo(int dato) {
      info=dato;
      anterior=null;
      posterior=null;
   }

}
```

***NOTA: Solo se debe aumentar en la clase lista la función, esto es una lista enlazada doble**

1. Cree una función que retorne verdadero si un elemento N existe en la lista y caso contrarioretorne falso en una lista enlazada doble.

```
public boolean buscar(int dato){
      Nodo aux=Inicio;
      while(aux!=null){
         if(aux.info==dato)
            return true;
         aux=aux.posterior;
      }
      return false;
   }
```

2. Cree una función que elimine un elemento de información N de una lista enlazada doble.

```
public void eliminar(int num){
      Nodo aux=Inicio;
      if(aux.info==num){
         Inicio=aux.posterior;
         Inicio.anterior=null;
         return;
      }
      while(aux!=null){
         if(aux.info==num){
            if(aux.posterior==null){
               aux.anterior.posterior=null;
               ultimo=aux.anterior;
            }else{
               aux.anterior.posterior=aux.posterior;
               aux.posterior.anterior=aux.anterior;
            }
            return;
```

```
        }
        aux=aux.posterior;
      }
    }
```

3. Realice una función que reciba como parámetro un elemento N, la función debe retornar la posición en la que se encuentra y si no existe la función retornara -1 en una lista enlazada doble.

```
public int pos(int dato){
      int pos=0;
      boolean si=false;
      Nodo aux=Inicio;
      while(aux!=null){
        if(aux.info==dato){
          return pos;
        }
        pos++;
        aux=aux.posterior;
      }
      return -1;
    }
```

4. Realice una función que invierta una lista enlazada doble.

```
public void invertir(){
      Nodo aux1=Inicio;
      Nodo aux2=ultimo;
      for(int i=0;i<tam/2;i++){
        int auxa=aux1.info;
        int auxb=aux2.info;
        aux1.info=auxb;
        aux2.info=auxa;
        aux1=aux1.posterior;
        aux2=aux2.anterior;
      }
    }
```

5. Realice una función que retorne la suma de los elementos que sean palíndromos de una lista enlazada doble.

```
public int sumapalindromo(){
      Nodo aux=Inicio;
      int suma=0;
      while(aux!=null){
        if(palindromo(aux.info))
          suma=suma+aux.info;
        aux=aux.posterior;
      }
      return suma;
    }
    public boolean palindromo(int num){
      int numero=num;
```

```
    int invertido=0;
    while(numero!=0) {
       int digito=numero%10;
       invertido=invertido*10+digito;
       numero=numero/10;
    }
    if(num==invertido)
       return true;
    return false;
  }
```

6. Crear una lista circular doble y realizar un menú donde pueda hacer lo siguiente:
   1) Menú
   2) Insertar al inicio
   3) Insertar al final
   4) Obtener tamaño de la lista
   5) Mostrar lista de inicio a fin
   6) Mostrar lista de fin a inicio
   7) Eliminar nodo inicial
   8) Eliminar nodo final
   9) Contar pares
   10) Mostrar primos
   11) Sumar elementos
   12) Calcular promedio
   13) Mostrar números mayores al promedio
   14) Insertar un elemento en una posición N
   15) Buscar elemento y mostrar su posición
   16) Invertir lista
   17) Eliminar todos los elementos de la lista
   18) Ordenar en orden descendente
   19) Rotar n veces a lado derecho
   20) Salir

Clase Main

```
public static void main(String[] args) {
    // TODO code application logic here
    Lista nuevo=new Lista();
    Scanner ie=new Scanner(System.in);
    boolean aux=true;
    while(aux==true){
       menu();
       int x=ie.nextInt();
       int dato;
       switch (x) {
          case 1:
             menu();
             break;
```

```
case 2:
   System.out.print("Ingrese Dato: ");
   dato=ie.nextInt();
   nuevo.insertarI(dato);
   break;
case 3:
   System.out.print("Ingrese Dato: ");
   dato=ie.nextInt();
   nuevo.insertarU(dato);
   break;
case 4:
   System.out.println("El tamaño es: "+nuevo.tamaño());
   break;
case 5:
    nuevo.mostrarIF();
   break;
case 6:
   nuevo.mostrarFI();
   break;
case 7:
   nuevo.eliminarInicio();
   break;
case 8:
   nuevo.eliminarFinal();
   break;
case 9:
   System.out.println(nuevo.contarPares());
   break;
case 10:
   nuevo.Primos();
   break;
case 11:
   System.out.println(nuevo.sumando());
   break;
case 12:
   System.out.println(nuevo.promedio());
   break;
case 13:
   nuevo.mostrarprom();
   break;
case 14:
   System.out.print("Valor a Insertar :");
   dato=ie.nextInt();
   System.out.println("En que posicion desea insertar: ");
```

```
            int pos=ie.nextInt();
            nuevo.InsertarN(pos, dato);
            break;
         case 15:
            System.out.print("Que numero quiere buscar:");
            dato=ie.nextInt();
            nuevo.Buscar(dato);
            break;
         case 16:
            nuevo.invertir();
            break;
         case 17:break;
         case 18:break;
         case 19:break;
         case 20:
            System.out.println("Gracias por su participacion");
            aux=false;
            break;
         default:
            System.out.println("Opcion no valida intente nuevamente");
            break;
      }
   }
}
static void menu(){
   System.out.println("1. Menu");
   System.out.println("2. Insertar Inicio");
   System.out.println("3. Insertar Final");
   System.out.println("4. Obtener tamaño de la lista");
   System.out.println("5. Mostrar Lista de Inicio a Fin");
   System.out.println("6. Mostrar Lista de Fin a Inicio");
   System.out.println("7. Eliminar Nodo Inicial");
   System.out.println("8. Eliminar Nodo Final");
   System.out.println("9. Contar Pares");
   System.out.println("10. Mostrar Primos");
   System.out.println("11. Sumar Elementos");
   System.out.println("12. Calcular Promedio");
   System.out.println("13. Mostrar Numero Mayores que el Promedio");
   System.out.println("14. Insertar un Elemento en una Posicion N");
   System.out.println("15. Buscar elemento y mostrar su posicion");
   System.out.println("16. Invertir Lista");
   System.out.println("17. Eliminar Todos los Elementos de la Lista");
   System.out.println("18. Ordenar por Orden Descendente");
   System.out.println("19. Rotar N veces al Lado Derecho");
```

```
        System.out.println("20. Salir");
    }
```

Clase Nodo

```
public class Nodo {
    int info;
    Nodo anterior;
    Nodo posterior;
    public Nodo(int info) {
        this.info = info;
        anterior=null;
        posterior=null;
    }
}
```

Clase Lista

```
public class Lista {
    Nodo Inicio;
    Nodo ultimo;
    int tam;
    public Lista(){
        tam=0;
    }
    //Insertar al Inicio
    public void insertarI(int dato){
        Nodo aux=new Nodo(dato);
        if(tam==0){
            Inicio=aux;
        }else{
            Inicio.anterior=aux;
            aux.posterior=Inicio;
        }
        Inicio=aux;
        Inicio.anterior=ultimo;
        ultimo.posterior=Inicio;
        tam++;
    }
    //Insertar al Final
    public void insertarU(int dato){
        Nodo aux=new Nodo(dato);
        if(tam==0){
            Inicio=aux;
        }else{
            ultimo.posterior=aux;
            aux.anterior=ultimo;
            aux.posterior=Inicio;
            Inicio.anterior=aux;
        }
        ultimo=aux;
```

```
   tam++;
}
//Mostrar Tamaño
public int tamaño(){
   return tam;
}
//mostrar de inicio -> final
public void mostrarIF(){
   Nodo aux=Inicio;
   for(int i=0;i<10;i++){
      System.out.println(aux.info);
      aux=aux.posterior;
   }
}
// mostrar final -> inicio
public void mostrarFI(){
   Nodo aux=ultimo;
   for(int i=0;i<tam;i++){
      System.out.println(aux.info);
      aux=aux.anterior;
   }
}
//Eliminar el dato del inicio
public void eliminarInicio(){
   if(tam==1){
      Inicio=null;
      tam--;
      return;
   }else{
      ultimo.posterior=Inicio.posterior;
      Inicio.posterior.anterior=ultimo;
      tam--;
      Inicio=Inicio.posterior;
   }
}
//eliminar el dato del final
public void eliminarFinal(){
   if(tam==1){
      Inicio=null;
      ultimo=null;
      tam--;
      return;
   }else{
      ultimo.anterior.posterior=Inicio;
      Inicio.anterior=ultimo.anterior;
      ultimo=ultimo.anterior;
      tam--;
   }
```

```
}
//contar pares
public int contarPares(){
   Nodo aux=Inicio;
   int cont=0;
   for(int i=0;i<tam;i++){
      if(aux.info%2==0)
         cont++;
      aux=aux.posterior;
   }
   return cont;
}
//mostrar los primos
public void Primos(){
   Nodo aux=Inicio;
   for(int i=0;i<tam;i++){
      if(SioNoPrimo(aux.info))
         System.out.println(aux.info);
      aux=aux.posterior;
   }
}
public boolean SioNoPrimo(int num){
   for(int i=2;i<Math.sqrt(num);i++){
      if(num%i==0)
         return false;
   }
   return true;
}
//sumar los elementos
public int sumando(){
   Nodo aux=Inicio;
   int suma=0;
   for(int i=0;i<tam;i++){
      suma=suma+aux.info;
      aux=aux.posterior;
   }
   return suma;
}
//promedio
public float promedio(){
   int sum=sumando();
   return sum/tam;
}
//mostrar numeros mayores al promedio
public void mostrarprom(){
   Nodo aux=Inicio;
   float auxp=promedio();
   for(int i=0;i<tam;i++){
```

```
        if(aux.info>=auxp)
            System.out.println(aux.info);
        aux=aux.posterior;
    }
}
//Insertar en una posicion
public void InsertarN(int pos1,int dato){
    Nodo aux=Inicio;
    Nodo nuevo=new Nodo(dato);
    int pos=0;
    float auxp=promedio();
    for(int i=0;i<tam;i++){
        if(pos==pos1){
            aux.anterior.posterior=nuevo;
            nuevo.anterior=aux.anterior;
            aux.anterior=nuevo;
            nuevo.posterior=aux;
            if(pos==tam)
                ultimo=nuevo;
            tam++;
        }
        pos++;
        aux=aux.posterior;
    }
}
//Buscar un ELemento y mostrar su posicion
public void Buscar(int dato){
    Nodo aux=Inicio;
    int pos=0;
    float auxp=promedio();
    for(int i=0;i<tam;i++){
        if(aux.info==dato){
            System.out.println(pos);
            return;
        }
        pos++;
        aux=aux.posterior;
    }
    System.out.println(-1);
}
//Invertir
public void invertir(){
    Nodo aux1=Inicio;
    Nodo aux2=ultimo;
    for(int i=0;i<tam/2;i++){
        int auxa=aux1.info;
        int auxb=aux2.info;
        aux1.info=auxb;
```

```
            aux2.info=auxa;
            aux1=aux1.posterior;
            aux2=aux2.anterior;
        }
    }
}
```

Nota: La práctica debe ser entregada el día miércoles 29 de mayo a horas 14:00.