



**Universidad Tecnológica
del Norte de Guanajuato**
Organismo Público Descentralizado del Gobierno del Estado
“Educación y progreso para la vida”

TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

Licenciatura en Ingeniería en Tecnologías de la Información e Innovación Digital
(DSM)

ESTRUCTURA DE DATOS

UNIDAD II

R1

Grupo: GTID0141

Alumno: Avalos Melado Rey Gerardo

No.Control: 1224100509

Docente: Gabriel Barron Rodriguez

Lista simple enlazada

Rey Gerardo Ruíz Mellado.

Implementación de una lista simplemente enlazada con nodos en ^{aparte} tipo ^{de}

2-Corregir las siguientes clases en Java.
Nodo y ListaEnlazada.

Clase Nodo

```
Public class Nodo<T> {  
    private T dato;           // Encapsulamiento  
    private Nodo<T> siguiente; // Encapsulamiento
```

```
    Public Nodo(T dato) {  
        this.dato = dato;  
        this.siguiente = null;  
    }
```

// Getters y setters

```
    Public T getData() {  
        return dato;  
    }
```

```
    Public void setDato(T dato) {  
        this.dato = dato;  
    }
```

```
    Public Nodo<T> getSiguiente() {  
        return siguiente;  
    }
```

```
    Public void setSiguiente(Nodo<T> siguiente) {  
        this.siguiente = siguiente;  
    }
```

- ↓ Necesitas dedicar el doble de memoria
- ↓ Ejercicios de Punto de vista de tipos y ordenamiento
- ↓ Evitarás tener que ordenar los objetos de lista
- ↓ Tenerás la semana lista para las listas
- ↓ Dejados

Public class Nodo<T> {

 private T dato;

 private Nodo<T> siguiente;

 private Nodo<T> anterior;

 public Nodo<T> dato) {

 this.dato = dato;

 this.siguiente = null;

 this.anterior = null;

}

 public T getDato() {

 return dato;

}

 public void setDato(T dato) {

 this.dato = dato;

}

 public Nodo<T> getSiguenete() {

 return siguiente;

}

 public void setSiguenete(Nodo<T>)

 this.siguenete = siguiente;

}

 public Nodo<T> getAnterior() {

 return anterior;

}

 public void setAnterior(Nodo<T>)

 this.anterior = anterior;

}

}

```
Public class ListaDoble<T> {
    private Nodo<T> cabecera;
    private Nodo<T> cola;
}

Public ListaDoble<T>() {
    this.cabecera = null;
    this.cola = null;
}

// Agregar al inicio
Public void agregarAlInicio(T dato) {
    Nodo<T> Nuevo = new Nodo<T>(dato);
    If (Cabecera == null) {
        cabecera = Nuevo;
        cola = Nuevo;
        cola = Nuevo;
    } else {
        Nuevo.setSiguiente(Cabecera);
        Cabecera.setAnterior(Nuevo);
        cabecera = Nuevo;
    }
}

B
```

```
2 // Eliminar al final
Public void eliminarAlFinal() {
    If (Cola == null) {
        System.out.println("La lista está vacía");
        return;
    }
    If (cabecera == Cola) {
        cabecera = null;
        Cola = null;
    }
}
```

Norma

```
cda = cdg.getAnterior();  
cdg.setSiguiente(null);
```

3

3

// Imprimir hacia adelante

```
public void imprimirLista() {  
    Nodo <T> actual = cabecera;  
    System.out.println("Lista(Adelante) : ");  
    while (actual != null) {  
        System.out.print("Global.getData() -> ");  
        actual = actual.getAnterior();  
    }  
    System.out.println("null")  
}
```

3

3

Desarrollo

```
public void agregarAlFinal(T dato) {  
    Nodo <T> nuevo = new Nodo <T>(dato);  
    if (cabecera == null) {  
        cabecera = nuevo;  
    }  
    return
```

3

Nodo <T> actual = cabecera;

```
while (actual.getSiguiente() != null) {  
    actual = actual.getSiguiente();
```

3

```
actual.setSiguiente(nuevo);
```

3

3

Parte 4

public class Main {

 public static void main (String [] args) {

 ListaEnlazada < Integer > historial = new ListaEnlazada < Integer > ();

 System.out.println ("\" agregando transacciones\"");

 historial.agregarAlInicio (150);

 historial.agregarAlFinal (45);

 historial.agregarAlFinal (200);

 historial.agregarAlFinal (15);

 System.out.println ("\" Historial de transacciones después de agregar: \"");

 historial.imprimirLista ();

 System.out.println ("\" Cancelando la transacción más reciente (15) \"");

 historial.eliminarAlFinal ();

 System.out.println ("\" Historial de transacciones después de la cancelación \"");

 historial.imprimirLista ();

 }

}