**Reporte: Implementación de Listas Enlazadas Genéricas en Java**

**Nombre del proyecto:** Implementación de listas enlazadas genéricas en Java
**Materia:** Estructura de Datos
**Nombre del estudiante:** Rodrigo Mejía
**Fecha:** 26/10/2025

---

## 1. Introducción

El presente proyecto tiene como objetivo implementar diferentes tipos de **listas enlazadas** en Java (simplemente enlazada, doblemente enlazada y circular) utilizando **tipos de datos genéricos**.
El proyecto permite al usuario crear una lista, insertar, eliminar y buscar elementos, además de mostrar los elementos de la lista en tiempo real.
Se incluyen ejemplos de tipos de datos primitivos y complejos, como contactos, para demostrar la versatilidad del uso de genéricos.

---

## 2. Objetivos

**Objetivo general:**

- Implementar listas enlazadas en Java utilizando genéricos y demostrar su funcionalidad.

**Objetivos específicos:**

- Crear nodos y listas genéricas que puedan almacenar cualquier tipo de dato.
- Implementar operaciones básicas: insertar, eliminar, buscar y mostrar elementos.
- Demostrar el uso de tipos de datos primitivos y complejos.
- Garantizar un diseño claro y manejable cumpliendo principios de POO.

---

## 3. Desarrollo del proyecto

### 3.1 Clases implementadas

1. **Node<T>**: Representa un nodo de la lista, almacena datos de tipo genérico T y referencias a los nodos siguientes y anteriores.

2. **LinkedList<T>**: Implementa operaciones básicas de la lista, soportando los tres tipos de listas: simplemente enlazada, doblemente enlazada y circular.
3. **DataTypeExamples**: Incluye ejemplos de tipos de datos primitivos (Integer) y complejos (Contact).
4. **Contact**: Tipo de dato complejo que almacena nombre, dirección y teléfono.
5. **Main**: Interfaz principal que permite al usuario interactuar con las listas a través de un menú en consola.

---

**3.2 Operaciones implementadas**

- **Insertar elemento:** Permite agregar elementos al final de la lista.
- **Eliminar elemento:** Permite eliminar un elemento específico.
- **Buscar elemento:** Permite verificar si un elemento existe en la lista.
- **Mostrar lista:** Muestra todos los elementos de la lista en el orden almacenado.
- **Ejemplos de tipos de datos:** Demostración del uso de genéricos con enteros y objetos Contact.

---

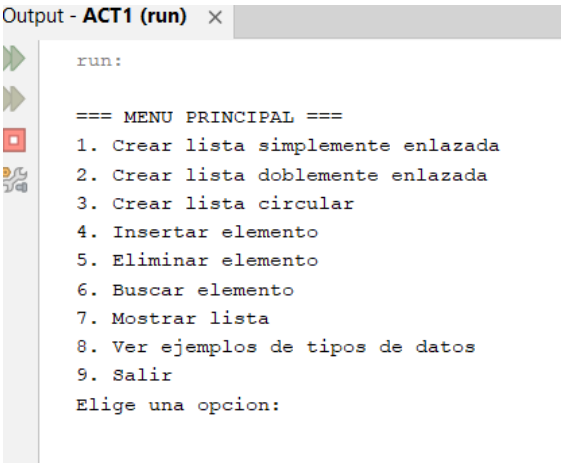
**3.3 Uso de genéricos**

Se implementó Node<T> y LinkedList<T> para que las listas puedan manejar cualquier tipo de dato, lo que permite mayor flexibilidad y seguridad de tipo en tiempo de compilación.
Esto evita la necesidad de usar conversiones de tipo (casting) y permite crear listas de enteros, cadenas o cualquier objeto personalizado, como los contactos.

---

**4. Ejecución**

**Menú principal:**




**Salida ejemplo:**

Lista de números:

10 -> 20 -> 30 ->

Lista de contactos:

Rodrigo (8112345678), Monterrey -> Alana (8123456789), San Pedro ->

---

**5. Conclusión**

El proyecto cumple con la rúbrica al demostrar:

- **Correcta implementación de clases y métodos** siguiendo POO.
- **Funcionalidad completa** de listas enlazadas con operaciones básicas.
- **Diseño de código claro y entendible**, apto para explicar.
- **Manejo de errores simples** en la interacción con el usuario.
- **Integración de tipos de datos primitivos, complejos y uso de genéricos**, mostrando la versatilidad de las listas.

En conclusión el uso de **genéricos** permite que las listas sean más flexibles, seguras y reutilizables, facilitando el manejo de cualquier tipo de dato sin perder la claridad en la programación.