Explicación del Código de Lista Simplemente Enlazada

Este código implementa una lista simplemente enlazada genérica en Java.

Declaración de la clase

```
public class ListaSimple<T extends Comparable<T>> implements IListaEnlazada<T> {
```

- Define una clase genérica `ListaSimple` que implementa la interfaz `IListaEnlazada`
- El tipo genérico `T` debe implementar `Comparable<T>` para permitir comparaciones entre elementos

Variables de instancia

```
private Nodo<T> cabeza;
private int tamanio;
```

- `cabeza`: Puntero al primer nodo de la lista
- `tamanio`: Contador del número de elementos en la lista

Constructor

```
public ListaSimple() {
    this.cabeza = null;
    this.tamanio = 0;
}
```

- Inicializa una lista vacía con cabeza nula y tamaño 0

Método agregarAlInicio

```
public void agregarAlInicio(T dato) {
    Nodo<T> nuevoNodo = new Nodo<>(dato);
    nuevoNodo.siguiente = cabeza;
    cabeza = nuevoNodo;
    tamanio++;
}
```

1. Crea un nuevo nodo con el dato proporcionado
2. Hace que el nuevo nodo apunte a la actual cabeza
3. Establece el nuevo nodo como la nueva cabeza
4. Incrementa el tamaño

Método agregarAlFinal

```
public void agregarAlFinal(T dato) {
    Nodo<T> nuevoNodo = new Nodo<>(dato);

    if (cabeza == null) {
        cabeza = nuevoNodo;
    } else {
        Nodo<T> actual = cabeza;
        while (actual.siguiente != null) {
            actual = actual.siguiente;
        }
        actual.siguiente = nuevoNodo;
    }
    tamanio++;
}
```

1. Crea un nuevo nodo
2. Si la lista está vacía, lo establece como cabeza
3. Si no, recorre hasta el último nodo y lo enlaza al nuevo nodo
4. Incrementa el tamaño

Método suprimir

```
public boolean suprimir(T dato) {
    if (cabeza == null) return false;

    // Eliminar cabeza
    if (cabeza.dato.equals(dato)) {
        cabeza = cabeza.siguiente;
        tamanio--;
        return true;
    }

    Nodo<T> actual = cabeza;
    while (actual.siguiente != null && !actual.siguiente.dato.equals(dato)) {
        actual = actual.siguiente;
    }

    if (actual.siguiente != null) {
        actual.siguiente = actual.siguiente.siguiente;
        tamanio--;
        return true;
    }

    return false;
}
```

1. Si la lista está vacía, retorna false
2. Si el dato está en la cabeza, elimina la cabeza
3. Busca el nodo anterior al que contiene el dato
4. Si lo encuentra, lo elimina reenlazando los nodos
5. Retorna true si eliminó, false si no encontró el dato

Método ordenar (Bubble Sort)

```
public void ordenar() {
    if (cabeza == null || cabeza.siguiente == null) return;

    boolean intercambio;
    do {
        intercambio = false;
        Nodo<T> actual = cabeza;

        while (actual.siguiente != null) {
            if (actual.dato.compareTo(actual.siguiente.dato) > 0) {
                // Intercambia datos
                T temp = actual.dato;
                actual.dato = actual.siguiente.dato;
                actual.siguiente.dato = temp;
                intercambio = true;
            }
            actual = actual.siguiente;
        }
    } while (intercambio);
}
```

1. Implementa el algoritmo Bubble Sort
2. Recorre la lista comparando elementos adyacentes
3. Si están en orden incorrecto, intercambia sus datos
4. Repite hasta que no haya más intercambios

Método listar

```
public String listar() {
    if (cabeza == null) return "Lista vacía";

    StringBuilder sb = new StringBuilder();
    Nodo<T> actual = cabeza;

    while (actual != null) {
        sb.append(actual.dato);
        if (actual.siguiente != null) sb.append(" -> ");
        actual = actual.siguiente;
    }
```

```
        return sb.toString();
    }
```

1. Si la lista está vacía, retorna mensaje
2. Recorre la lista concatenando los datos con "->" entre ellos
3. Retorna el string resultante

Métodos adicionales

```
public void vaciar() {
    cabeza = null;
    tamanio = 0;
}

public int getTamanio() {
    return tamanio;
}

public boolean estaVacia() {
    return cabeza == null;
}
```

- `vaciar()`: Reinicia la lista a estado vacío
- `getTamanio()`: Retorna el número de elementos
- `estaVacia()`: Retorna true si no hay elementos

El método `listarInverso()` está implementado pero vacío (retorna string vacío).