```
public class Lista<T> {

    // Classe interna para representar um nó na lista
    private class No {
        T dado;
        No proximo;

        No(T dado) {
            this.dado = dado;
            this.proximo = null;
        }
    }

    // Referência para o primeiro nó da lista
    private No primeiro;
    // Referência para o último nó da lista
    private No ultimo;
    // Contador de elementos na lista
    private int tamanho;

    // Construtor
    public Lista() {
        this.primeiro = null;
        this.ultimo = null;
        this.tamanho = 0;
    }

    // Método para adicionar um elemento ao final da lista
    public void adicionar(T dado) {
        No novoNo = new No(dado);
        if (primeiro == null) {
            primeiro = novoNo;
            ultimo = novoNo;
        } else {
            ultimo.proximo = novoNo;
            ultimo = novoNo;
        }
        tamanho++;
    }

    // Método para adicionar um elemento em uma posição específica na lista
    public void adicionar(T dado, int indice) {
        if (indice < 0 || indice > tamanho) {
            throw new IndexOutOfBoundsException("Índice fora dos limites da lista");
```

```
    }

    if (indice == 0) {
        No novoNo = new No(dado);
        novoNo.proximo = primeiro;
        primeiro = novoNo;
        if (tamanho == 0) {
            ultimo = novoNo;
        }
    } else if (indice == tamanho) {
        adicionar(dado);
    } else {
        No novoNo = new No(dado);
        No anterior = obterNo(indice - 1);
        novoNo.proximo = anterior.proximo;
        anterior.proximo = novoNo;
    }
    tamanho++;
}

// Método para remover um elemento da lista
public void remover(int indice) {
    if (indice < 0 || indice >= tamanho) {
        throw new IndexOutOfBoundsException("Índice fora dos limites da lista");
    }

    if (indice == 0) {
        primeiro = primeiro.proximo;
        if (primeiro == null) {
            ultimo = null;
        }
    } else {
        No anterior = obterNo(indice - 1);
        anterior.proximo = anterior.proximo.proximo;
        if (anterior.proximo == null) {
            ultimo = anterior;
        }
    }
    tamanho--;
}

// Método para obter um elemento da lista por índice
public T obter(int indice) {
    if (indice < 0 || indice >= tamanho) {
```

```
            throw new IndexOutOfBoundsException("Índice fora dos limites da lista");
        }

        return obterNo(indice).dado;
    }

    // Método para verificar se a lista está vazia
    public boolean vazia() {
        return tamanho == 0;
    }

    // Método para obter o tamanho da lista
    public int tamanho() {
        return tamanho;
    }

    // Método para limpar a lista
    public void limpar() {
        primeiro = null;
        ultimo = null;
        tamanho = 0;
    }

    // Método auxiliar para obter o nó de um índice específico
    private No obterNo(int indice) {
        No atual = primeiro;
        for (int i = 0; i < indice; i++) {
            atual = atual.proximo;
        }
        return atual;
    }

    // Método para imprimir os elementos da lista
    public void imprimir() {
        No atual = primeiro;
        while (atual != null) {
            System.out.print(atual.dado + " ");
            atual = atual.proximo;
        }
        System.out.println();
    }

    public static void main(String[] args) {
        Lista<Object> lista = new Lista<>();
```

```
        lista.adicionar(1);
        lista.adicionar("a");
        lista.adicionar(2.5);
        lista.adicionar("Texto");
        lista.imprimir(); // Saída esperada: 1 a 2.5 Texto
    }
}
```