



PECS | Programa de Pós-Graduação em  
Engenharia de Computação e Sistemas

# ALGORITMOS E ESTRUTURAS DE DADOS

Disciplina: ALGORITMOS E ESTRUTURAS DE DADOS

Carga Horária: 60h

Professor: **Dr. Reinaldo**

## Introdução

Estrutura Bidirecional - Uma lista duplamente encadeada é uma estrutura de dados que permite a navegação em ambas as direções, tanto para frente quanto para trás, através de nós que armazenam **dados** e ponteiros para o **próximo** e o nó **anterior**.

| Início   |   |       |   | Fim     |
|----------|---|-------|---|---------|
| 5        | ↔ | 8     | ↔ | 11      |
| Anterior | ↔ | Valor | ↔ | Próximo |

- Cada nó aponta para o nó **anterior** e para o nó **próximo**.
- O primeiro nó não tem um nó anterior, então o ponteiro anterior é **null**.
- O último nó não tem um nó próximo, então o ponteiro próximo é **null**.

## Exemplo - Duplamente Encadeada

| Início   |   |       |   |  | Fim     |
|----------|---|-------|---|--|---------|
| null     | ↔ | 5     | ↔ |  | 8       |
| Anterior | ↔ | Valor | ↔ |  | Próximo |

- Nó anterior aponta para **null** (início da lista).
- valor é 5.
- próximo aponta para o nó com valor 8.

| Início   |   |       |   |  | Fim     |
|----------|---|-------|---|--|---------|
| 8        | ↔ | 11    | ↔ |  | null    |
| Anterior | ↔ | Valor | ↔ |  | Próximo |

- Nó anterior aponta para nó com valor **8**.
- valor é 11.
- próximo aponta para o nó com valor **null**.

## Inserção do Novo Nó

Valor [6]

| Início   |   |  |       |   |  |         | Fim |    |
|----------|---|--|-------|---|--|---------|-----|----|
| 5        | ↔ |  | 6     | ↔ |  | 8       | ↔   | 11 |
| Anterior | ↔ |  | Valor | ↔ |  | Próximo | ↔   |    |

- O próximo do nó com valor 5 apontará para o novo nó.
- O anterior do nó com valor 8 apontará para o novo nó.
- O anterior do novo nó apontará para o nó com valor 5.
- O próximo do novo nó apontará para o nó com valor 8.

## Renover Nó

Vamos remover o nó com valor 8:

| Início   |   |  |       |   |                    |   | Fim     |
|----------|---|--|-------|---|--------------------|---|---------|
| 5        | ↔ |  | 6     | ↔ | <del>8</del>       | ↔ | 11      |
| Anterior | ↔ |  | Valor | ↔ | <del>Próximo</del> | ↔ | Próximo |

- O próximo do nó com valor 6 apontará para o nó com valor 11.
- O anterior do nó com valor 11 apontará para o nó com valor 6.

## Criação da Classe Lista.java

```
package estruturalistaligadas;

public class Listaencadeada {
    private Object elemento;
    private Listaencadeada proximo;
    private Listaencadeada anterior; //Inserção do Anterior

    public Listaencadeada(Listaencadeada proximo, Object elemento) {
        this.proximo = proximo;
        this.elemento = elemento; }

    public void setProximo(Listaencadeada proximo) {
        this.proximo = proximo; }


    public Listaencadeada getProximo() {
        return proximo; }




    public Object getElemento() {
        return elemento; }

    public Listaencadeada(Object elemento) {
        this(null, elemento); }

    public Listaencadeada getAnterior() {
        return anterior; }

    public void setAnterior(Listaencadeada anterior) {
        this.anterior = anterior; }
}
```

▼  estruturalistaligadas

- >  Listaencadeada.java
- >  Listaligadaduplamente.java
- >  Principal.java





## Criação da Classe Listaligadaduplamente.java

```
public class Listaligadaduplamente {

    private Listaencadeada primeira;
    private int totalDeElementos;
    private Listaencadeada ultima;

    public void adicionaNoComeco(Object elemento) {
        if(this.totalDeElementos == 0) {
            Listaencadeada nova = new Listaencadeada(elemento);
            this.primeira = nova;
            this.ultima = nova;
        } else {
            Listaencadeada nova = new Listaencadeada(this.primeira, elemento);
            this.primeira.setAnterior(nova);
            this.primeira = nova; }
        this.totalDeElementos++;}

    public void adiciona(Object elemento) {
        if(this.totalDeElementos == 0) {
            adicionaNoComeco(elemento);
        } else {
            Listaencadeada nova = new Listaencadeada(elemento);
            this.ultima.setProximo(nova);
            nova.setAnterior(this.ultima);
            this.ultima = nova;
            this.totalDeElementos++;} }

    public void adiciona(int posicao, Object elemento) {
        if(posicao == 0) {
            adicionaNoComeco(elemento);
        } else if (posicao == this.totalDeElementos) {
            this.adiciona(elemento);
        } else {
            Listaencadeada anterior = busca(posicao - 1);
            Listaencadeada proxima = anterior.getProximo();
            Listaencadeada nova = new Listaencadeada(anterior.getProximo(), elemento);
            nova.setAnterior(anterior);
            anterior.setProximo(nova);
            proxima.setAnterior(nova);
            this.totalDeElementos++;} }
```

estruturalistaligadas

- > Listaencadeada.java
- > Listaligadaduplamente.java
- > Principal.java







## Criação da Classe Listaligadaduplamente.java

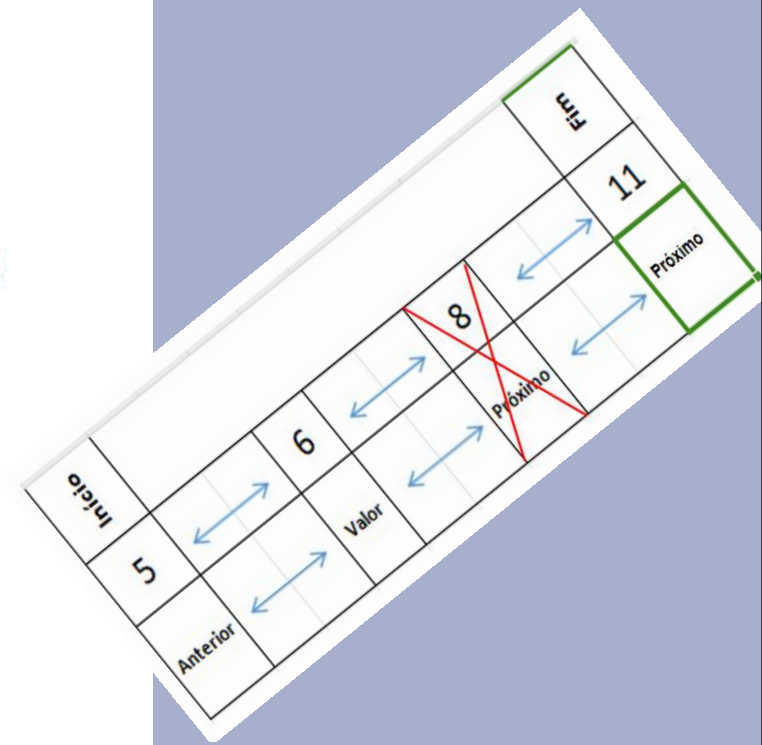
```
public void removeDoComeco() {
    if(this.totalDeElementos == 0) {
        throw new IllegalArgumentException("lista vazia"); }

    this.primeira = this.primeira.getProximo();
    this.totalDeElementos--;
    if(this.totalDeElementos == 0) {
        this.ultima = null; } }

public void removeDoFim() {
    if(this.totalDeElementos == 1) {
        this.removeDoComeco();
    } else {
        Listaencadeada penultima = this.ultima.getAnterior();
        penultima.setProximo(null);
        this.ultima = penultima;
        this.totalDeElementos--; } }

public void remove(int posicao) {
    if(posicao == 0) {
        this.removeDoComeco();
    } else if (posicao == this.totalDeElementos - 1) {
        this.removeDoFim();
    } else {
        Listaencadeada anterior = this.busca(posicao - 1);
        Listaencadeada atual = anterior.getProximo();
        Listaencadeada proxima = atual.getProximo();
        anterior.setProximo(proxima);
        proxima.setAnterior(anterior);
        this.totalDeElementos--; } }
```





▼  estruturalistaligadas  
 >  Listaencadeada.java  
 >  Listaligadaduplamente.java  
 >  Principal.java

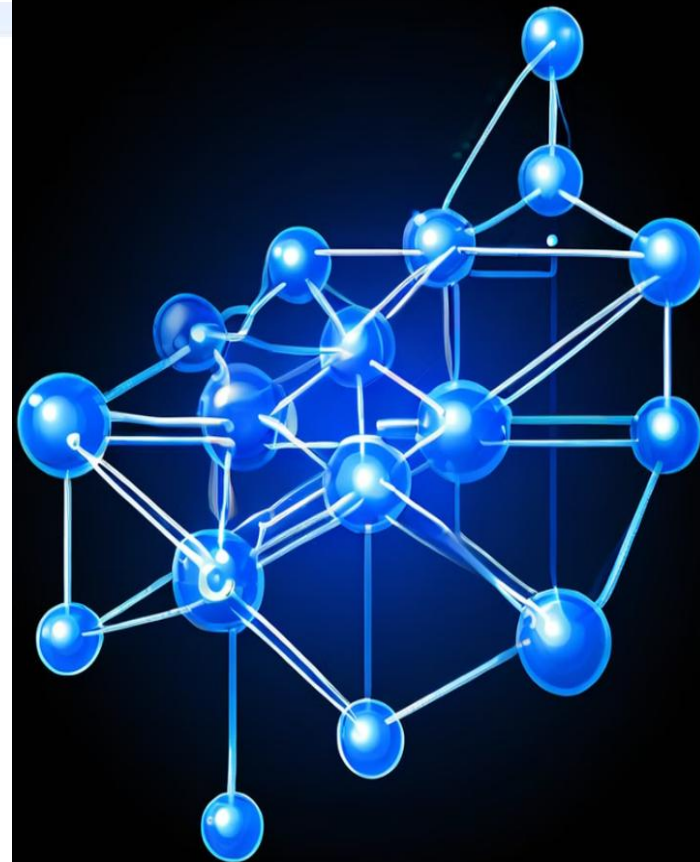




## Criação da Classe Listaligadaduplamente.java

```
public int tamanho() {  
    return this.totalDeElementos;  
}  
  
public boolean Localiza(Object elemento) {  
    Listaencadeada atual = this.primeira;  
  
    while(atual != null) {  
        if(atual.getElemento().equals(elemento)) {  
            return true;  
        }  
        atual = atual.getProximo();  
    }  
    return false;  
}  
  
@Override  
public String toString () {  
  
    if(this.totalDeElementos == 0) {  
        return "()"; }  
  
    Listaencadeada atual = primeira;  
    StringBuilder builder = new StringBuilder("(");  
    for(int i = 0; i < totalDeElementos; i++) {  
        builder.append(atual.getElemento());  
        builder.append(",");  
        atual = atual.getProximo();  
    }  
    builder.append(")");  
    return builder.toString(); }}
```

▼  estruturalistaligadas  
    >  Listaencadeada.java  
    >  Listaligadaduplamente.java  
    >  Principal.java



## Criação da Classe Principal.java

```
public class Principal {  
    public static void main(String[] args) {  
  
        Listaligadaduplamente lista = new Listaligadaduplamente();  
  
        System.out.println(lista);  
        lista.adicionaNoComeco("ANDERSON");  
        System.out.println(lista);  
        lista.adicionaNoComeco("LUAN");  
        System.out.println(lista);  
        lista.adicionaNoComeco("LORENA");  
        System.out.println(lista);  
        lista.adiciona("FRANK");  
        System.out.println(lista);  
        System.out.println(lista.tamanho());  
        System.out.println(lista);  
        lista.removeDoFim();  
        System.out.println(lista);  
        lista.adicionaNoComeco("ROGERIO");  
        System.out.println(lista);  
        lista.adiciona(3, "LUIS CARLOS");  
        System.out.println(lista);  
        lista.remove(2);  
        System.out.println(lista);  
        System.out.println (lista.Localiza("ROGERIO"));  
        System.out.println (lista.Localiza("REINALDO"));  
        System.out.println(lista);  
  
    }  
}
```

- ✚ estruturalistaligadas
- > Listaencadeada.java
- > Listaligadaduplamente.java
- > Principal.java

