# Aula anterior

- TAD linear: Pilha
- TAD Flexível Pilha

# Tipo Abstrato de Dados (Lista)

Prof. Diego Silva Caldeira Rocha

# Objetivos

- Introdução a Lista
- Tipos Abstratos de Dados Lineares: Lista
- Tipos Abstratos de Dados Flexível: Lista Simples
- Tipos Abstratos de Dados Flexível: Lista Dupla

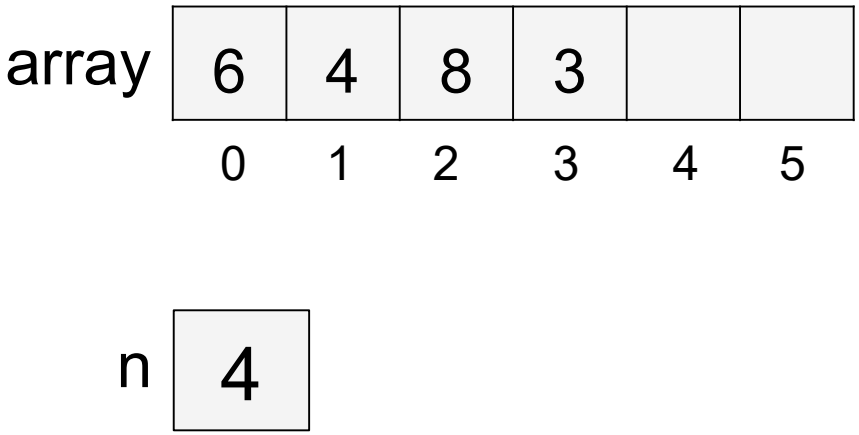
# Introdução

- As listas são um Tipo Abstrato de Dados (TAD) no qual podemos inserir e remover elementos em **qualquer posição**

- Exemplos:
  - Lista de valores (*array* de números inteiros)
  - Lista de nomes (*array* de strings)
  - Lista de notas (*array* de números reais)
  - Lista de carros (*array* de objetos do tipo carro)

# Variáveis da Lista

- *array* (de elementos) e n (contador)

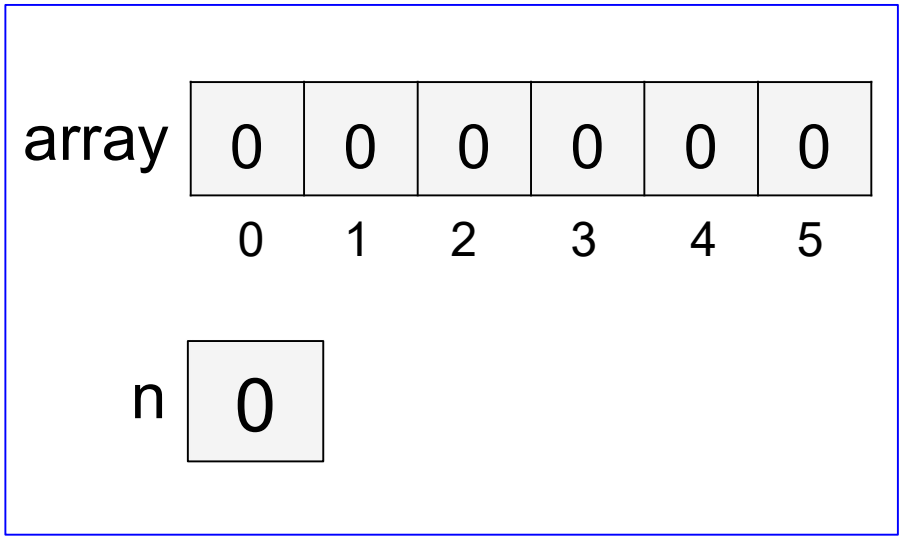


# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)    { ... }
    void inserirFim(int x)    {  ...  }
    void inserir(int x, int pos)    {  ...  }
    int removerInicio()    {  ...  }
    int removerFim()    {  ...  }
    int remover(int pos)    {  ...  }
    void mostrar () {  ...  }
}
```

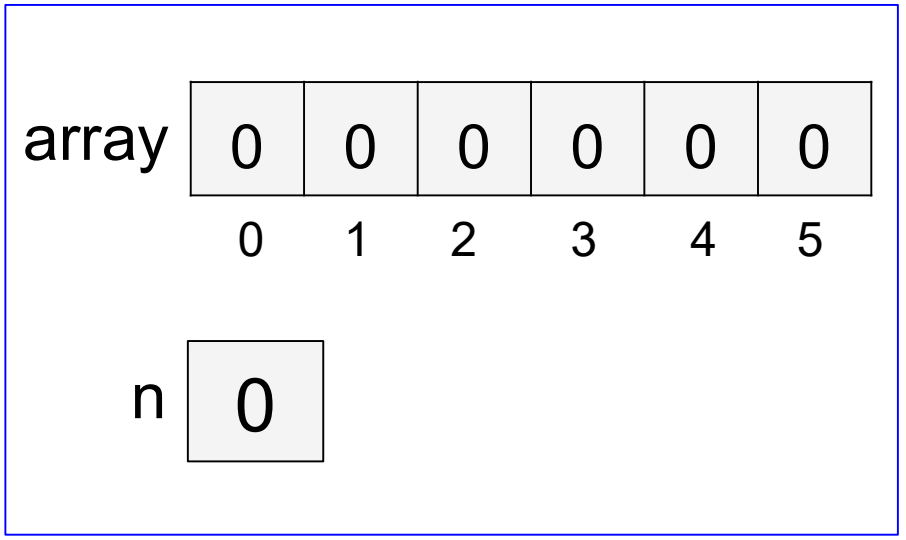
# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)    {  ...  }
    void inserirFim(int x)    {  ...  }
    void inserir(int x, int pos)    {  ...  }
    int removerInicio()    {  ...  }
    int removerFim()    {  ...  }
    int remover(int pos)    {  ...  }
    void mostrar () {  ...  }
}
```



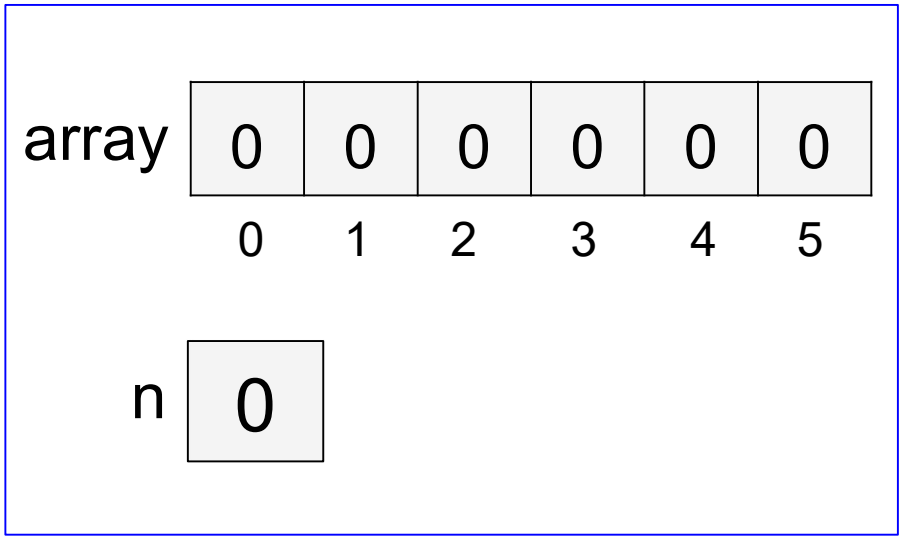
# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)   {  ...  }
    void inserirFim(int x)   {  ...  }
    void inserir(int x, int pos)   {  ...  }
    int removerInicio()   {  ...  }
    int removerFim()   {  ...  }
    int remover(int pos)   {  ...  }
    void mostrar () {  ...  }
}
```



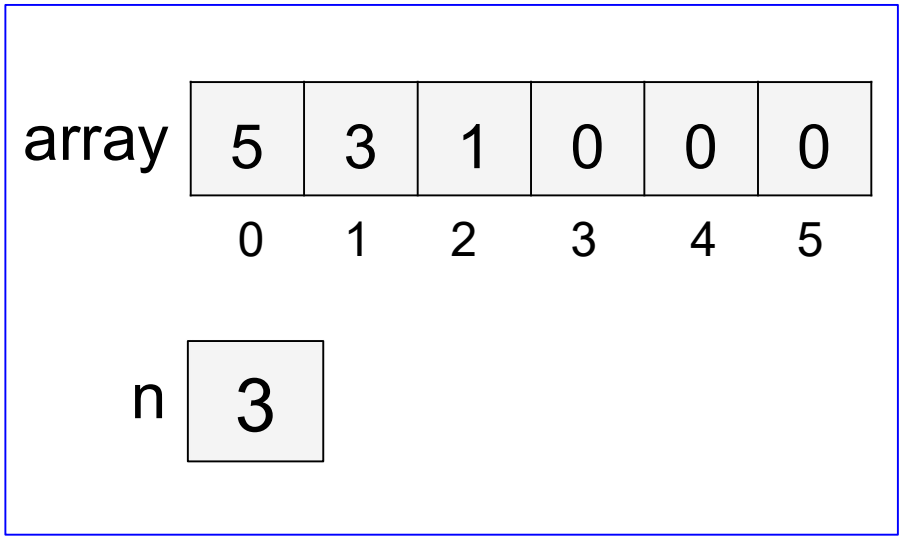
# Algoritmo em Java

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) throws Exception {

   if (n >= array.length)
      throw new Exception("Erro!");

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```



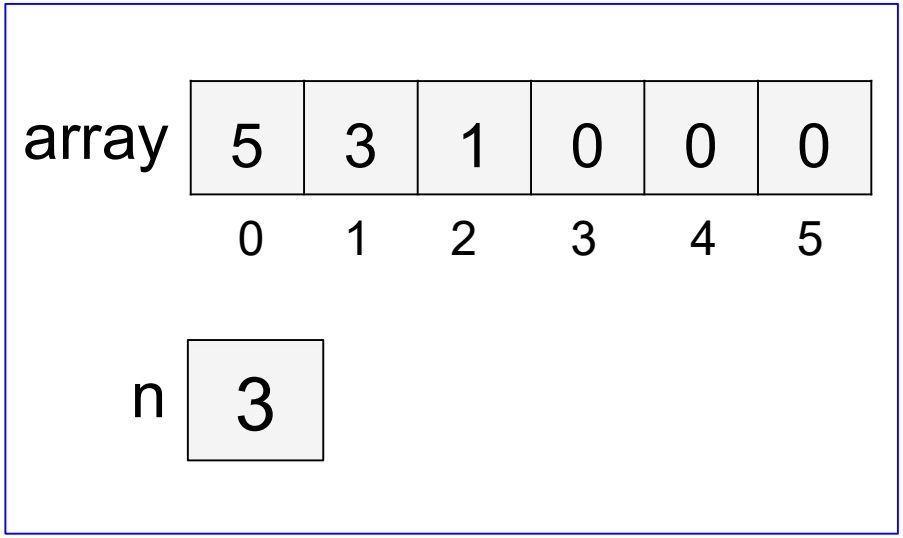
# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)   { ... }
    void inserirFim(int x)   { ... }
    void inserir(int x, int pos)   { ... }
    int removerInicio()   { ... }
    int removerFim()   { ... }
    int remover(int pos)   { ... }
    void mostrar () {  ... }
}
```



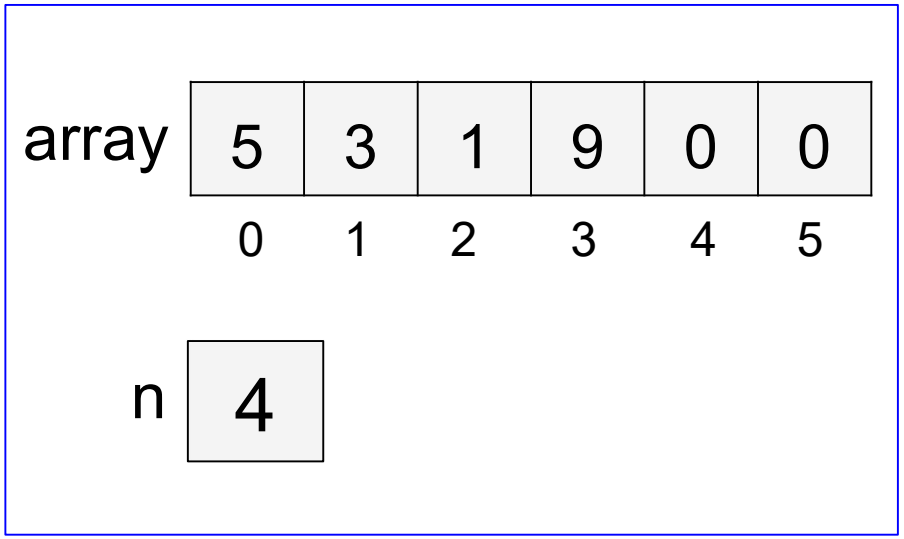
# Algoritmo em Java

```
// Exemplo: inserirFim(9)                                  x | 9
void inserirFim(int x) throws Exception {

   if (n >= array.length)
      throw new Exception("Erro!");

   array[n] = x;
   n++;
}
```



# Algoritmo em Java

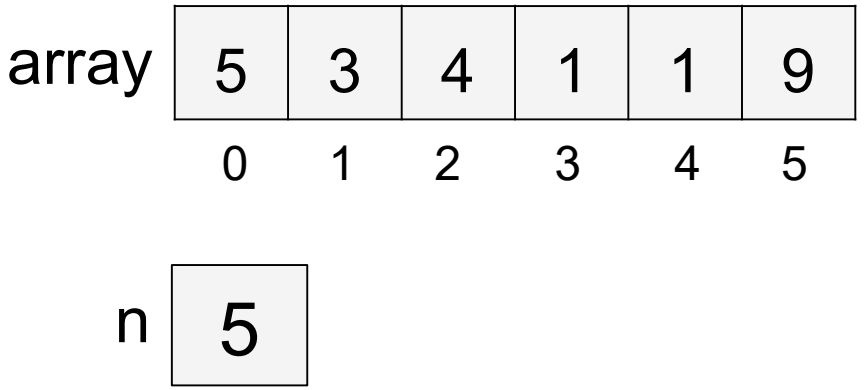
```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)    {  ...  }
    void inserirFim(int x)    {  ...  }
    void inserir(int x, int pos)    {  ...  }
    int removerInicio()    {  ...  }
    int removerFim()    {  ...  }
    int remover(int pos)    {  ...  }
    void mostrar () {  ...  }
}
```



# Algoritmo em Java

```
// Exemplo: inserir(7,2)
void inserir(int x, int pos) throws Exception {

 if (n >= array.length || pos < 0 || pos > n)
      throw new Exception("Erro!");

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
      array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

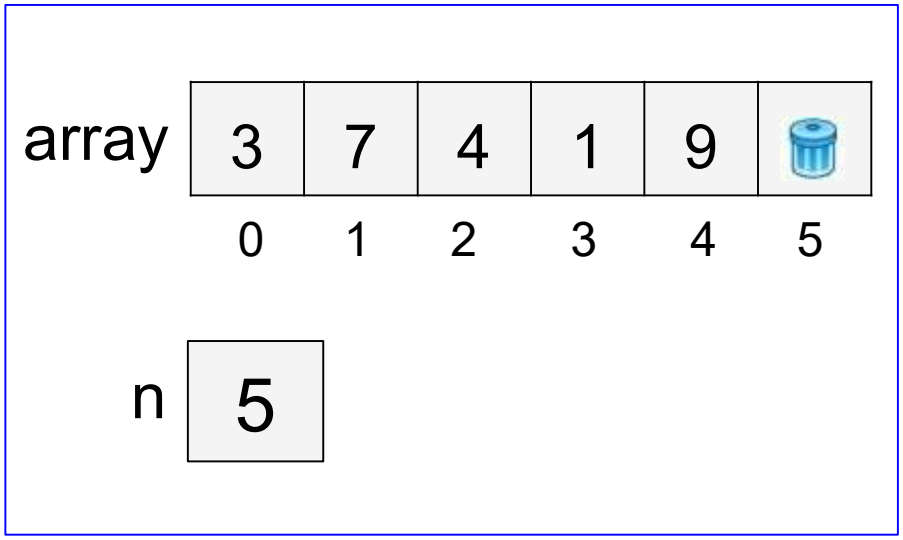


# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)   { ... }
    void inserirFim(int x)   { ... }
    void inserir(int x, int pos)   { ... }
    int removerInicio()   { ... }
    int removerFim()   { ... }
    int remover(int pos)   { ... }
    void mostrar () { ... }
}
```

# Algoritmo em Java

```
// Exemplo: removerFim()

int removerFim() throws Exception {
   if (n == 0)
      throw new Exception("Erro!");

   return array[--n];
}
```



# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {    this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)    {  ...  }
    void inserirFim(int x)    {  ...  }
    void inserir(int x, int pos)    {  ...  }
    int removerInicio()    {  ...  }
    int removerFim()
    int remover(int pos)    {  ...  }
    void mostrar () {  ...  }

}
```

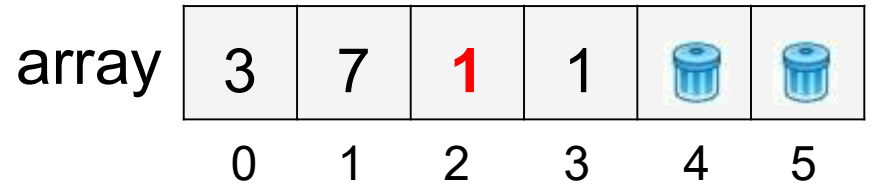
# Algoritmo em Java

```
// Exemplo: remover(2)

int remover(int pos) throws Exception {

   if (n == 0 || pos < 0 || pos >= n)
      throw new Exception("Erro!");

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 4

i: 2

array[2] <- array[3]



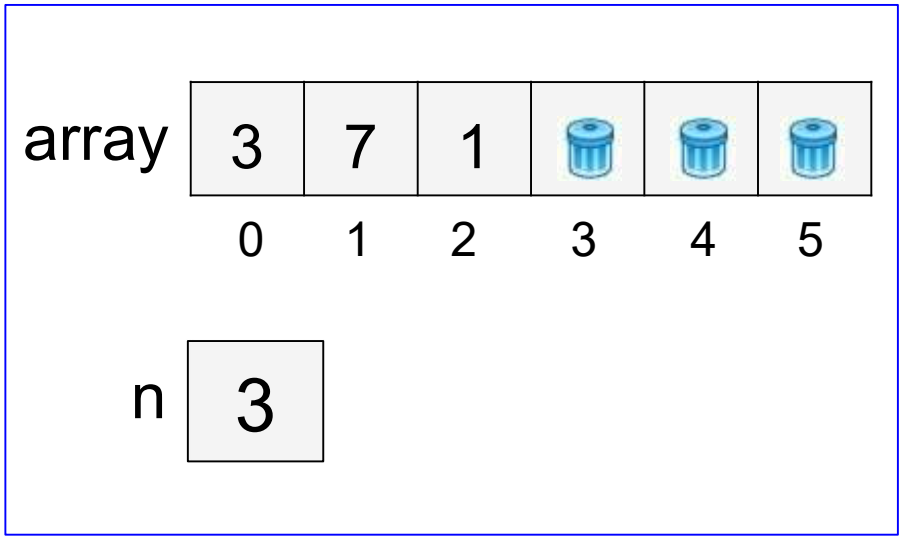

n: 3

# Algoritmo em Java

```
class Lista {
    int[] array;
    int n;

    Lista () {   this(6);  }
    Lista (int tamanho){
        array = new int[tamanho];
        n = 0;
    }

    void inserirInicio(int x)    { ... }
    void inserirFim(int x)    {  ...  }
    void inserir(int x, int pos)    {  ...  }
    int removerInicio()    {  ...  }
    int removerFim()    {  ...  }
    int remover(int pos)    {  ...  }
    void mostrar () {  ...  }
}
```

# Algoritmo em Java

```
void mostrar (){
   System.out.print("[ ");
   for (int i = 0; i < n; i++){
      System.out.print(array[i] + "  ");
   }
   System.out.println(" ]");
}
```




**Exercício**: O que será mostrado na tela?

Tela

# Código Fonte

- PrincipalLista.java, igual ao da estrutura sequencial

- Lista.java, tem os atributos primeiro e último e os métodos abaixo:

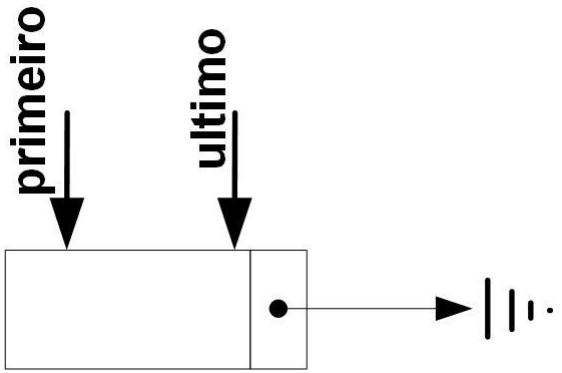
  - Inserir no início
  - Inserir no fim
  - Inserir
  - Remover no início
  - Remover no fim
  - Remover

# Classe Lista Simples

```
class Lista {
    private Celula primeiro, ultimo;
    public Lista () {
        primeiro = new Celula();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Classe Lista Simples

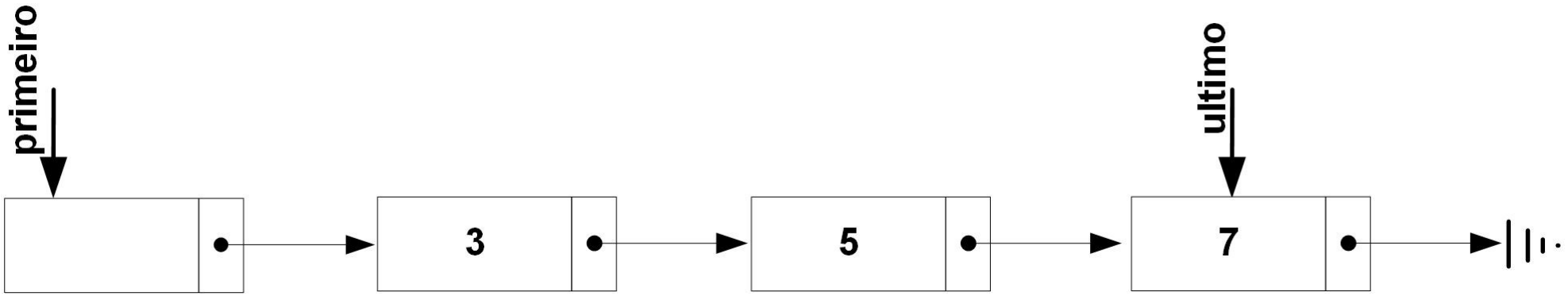
```
class Lista {
    private Celula primeiro, ultimo;
    public Lista () {
        primeiro = new Celula();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```



# Classe Lista Simples

```
class Lista {
    private Celula primeiro, ultimo;
    public Lista () {
        primeiro = new Celula();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

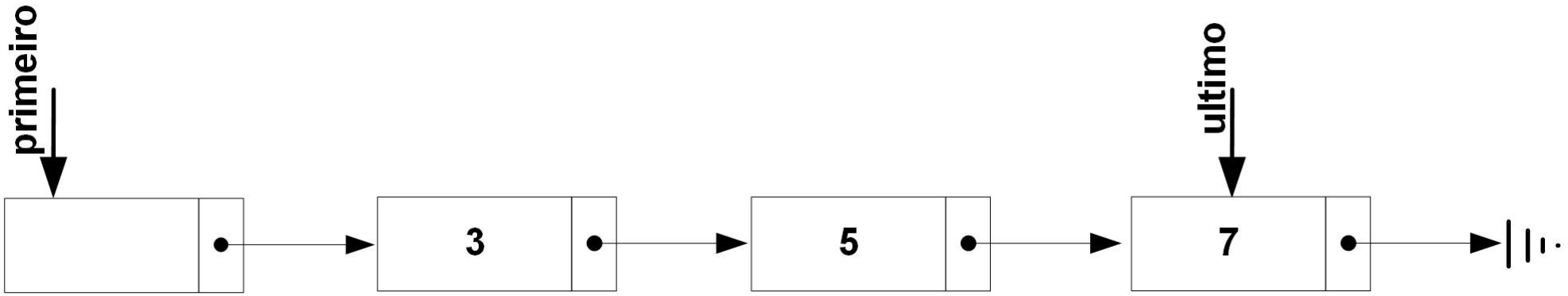
Iguais aos métodos da fila



# Classe Lista Simples

```
class Lista {
    private Celula primeiro, ultimo;
    public Lista () {
        primeiro = new Celula();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

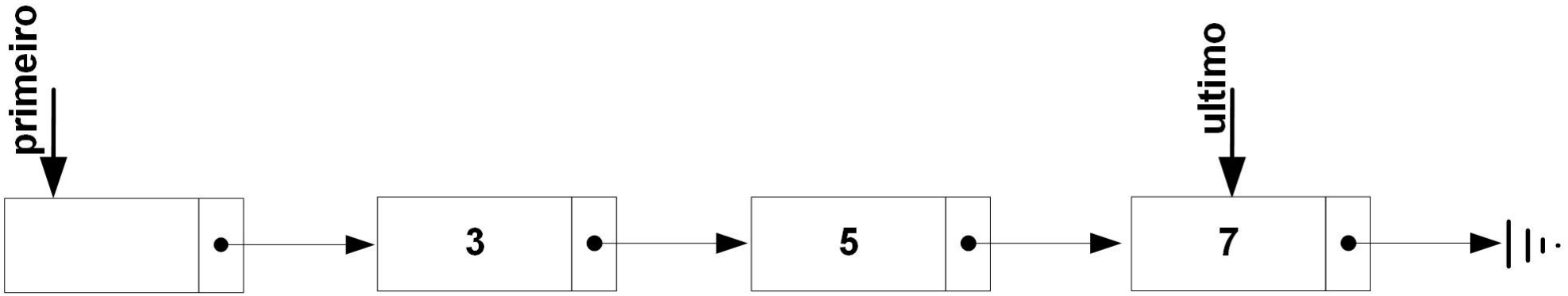
Igual aos da fila/pilha



# Classe Lista Simples

```
class Lista {
    private Celula primeiro, ultimo;
    public Lista () {
        primeiro = new Celula();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

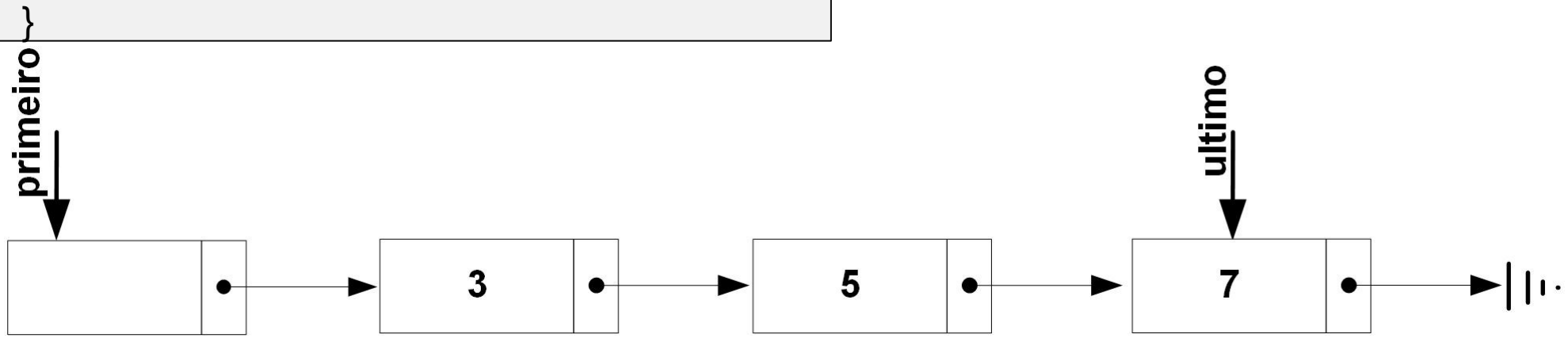
Assim, ...



# Classe Lista Simples

```
class Lista {

    public void inserirInicio(int x) { ... }

    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
```

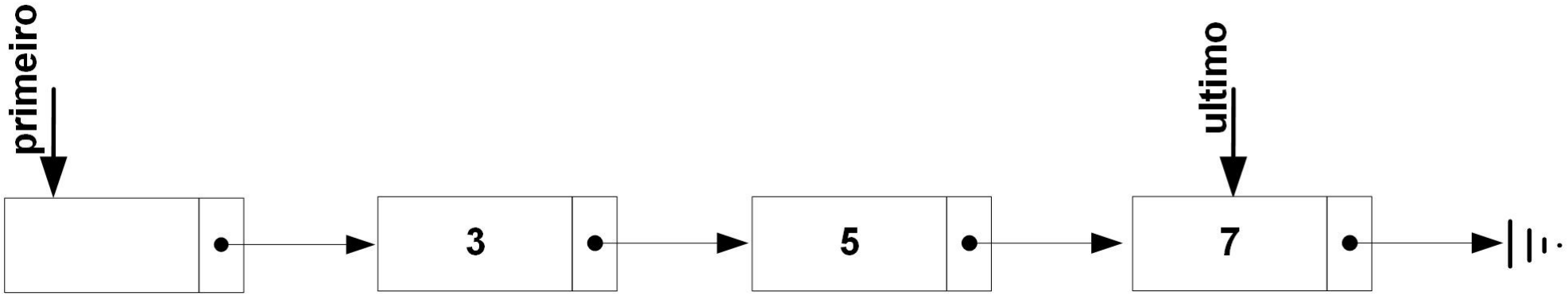
Assim, ...



# Classe Lista Simples

```
class Lista {
        ...
        public void inserirInicio(int x) { ... }
        public int removerFim() { ... }
        public void inserir(int x, int pos) { ... }
        public int remover(int pos) { ... }
}
```

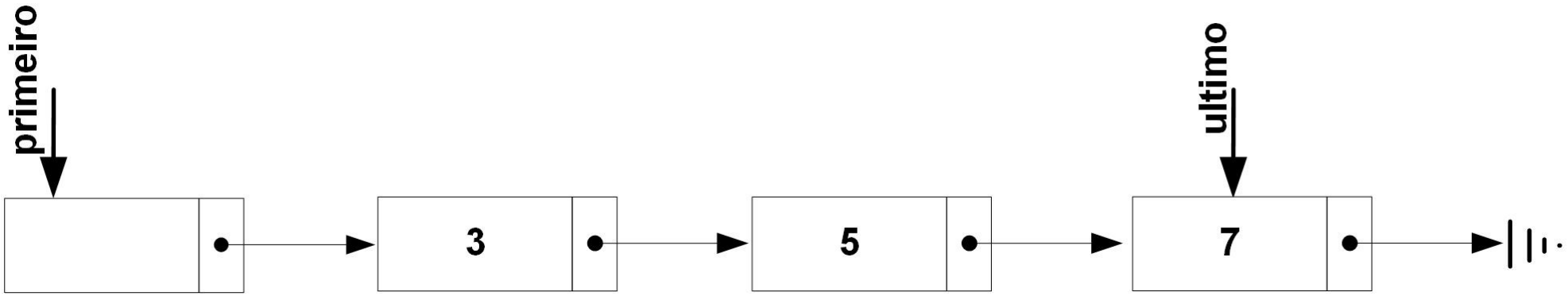
Assim, ...



# Inserir no Início

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

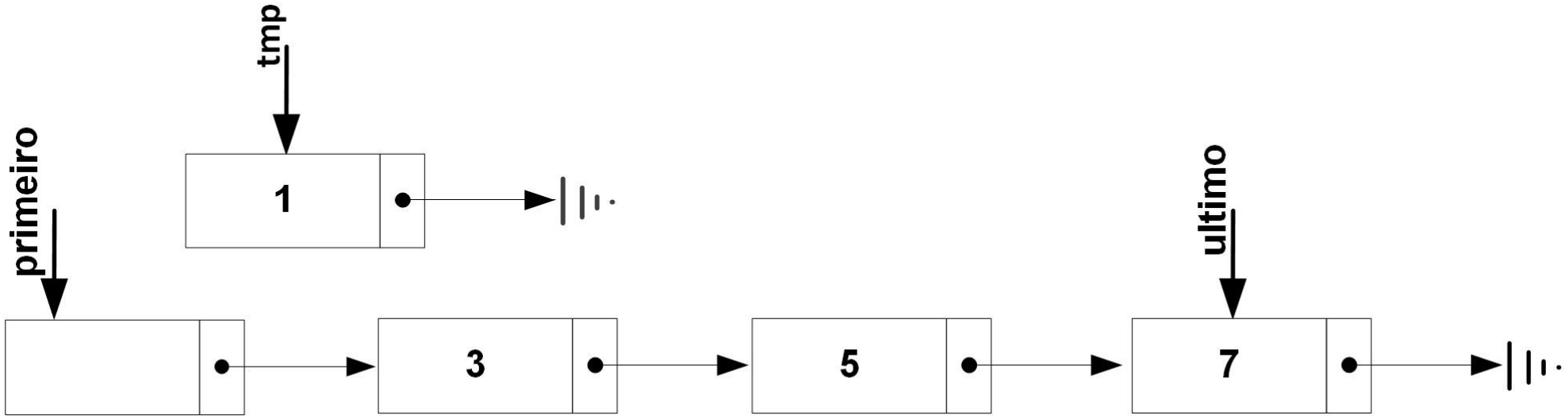
```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
       ultimo = tmp;
    }
    tmp = null;
}
```



# Inserir no Início

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

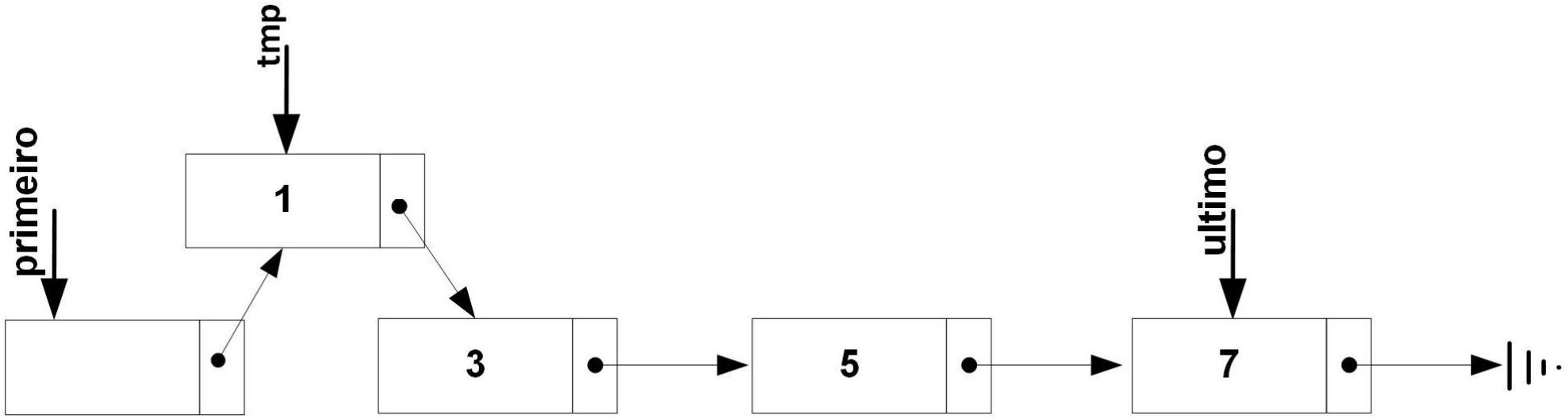
```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
       ultimo = tmp;
    }
    tmp = null;
}
```



# Inserir no Início

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
       ultimo = tmp;
    }
    tmp = null;
}
```

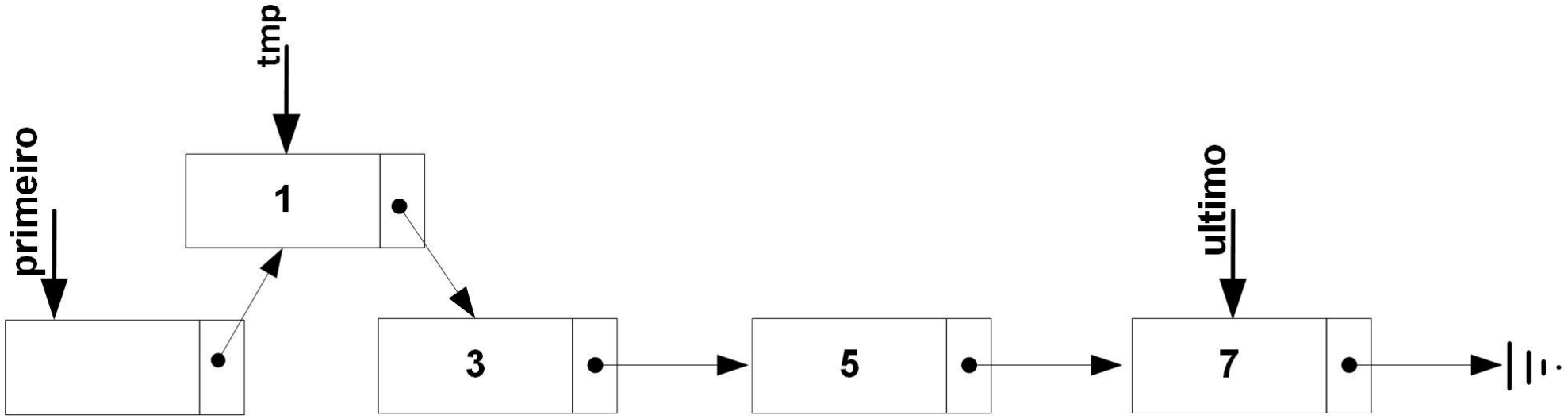


# Inserir no Início

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
        ultimo = tmp;
    }
    tmp = null;
}
```

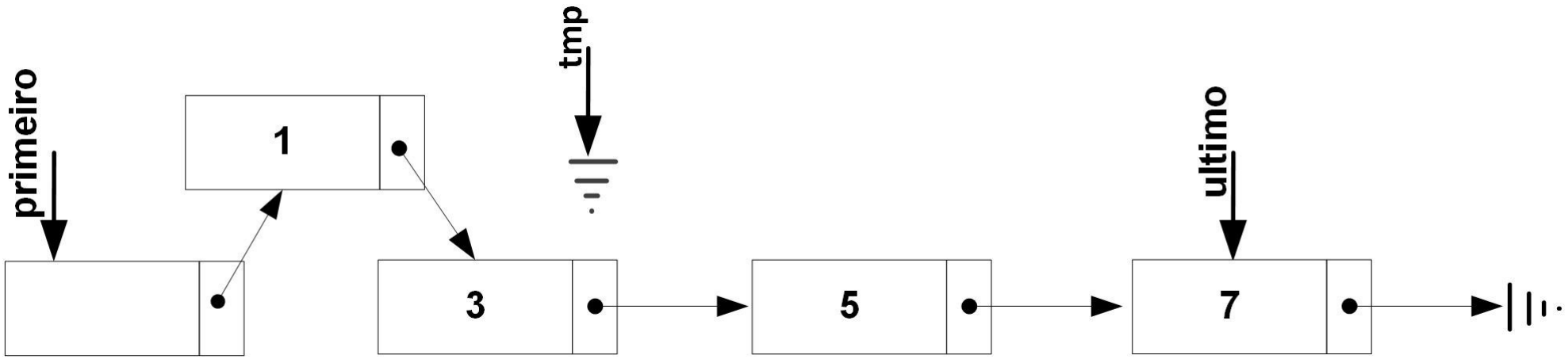
false



# Inserir no Início

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
        ultimo = tmp;
    }
    tmp = null;
}
```

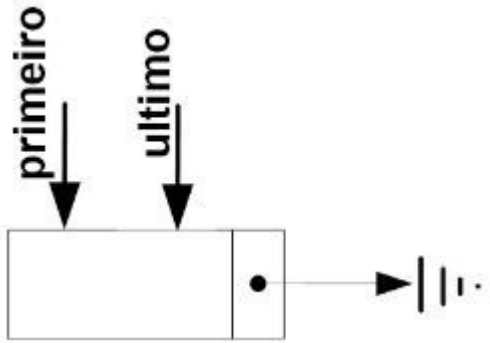


# Exercício

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

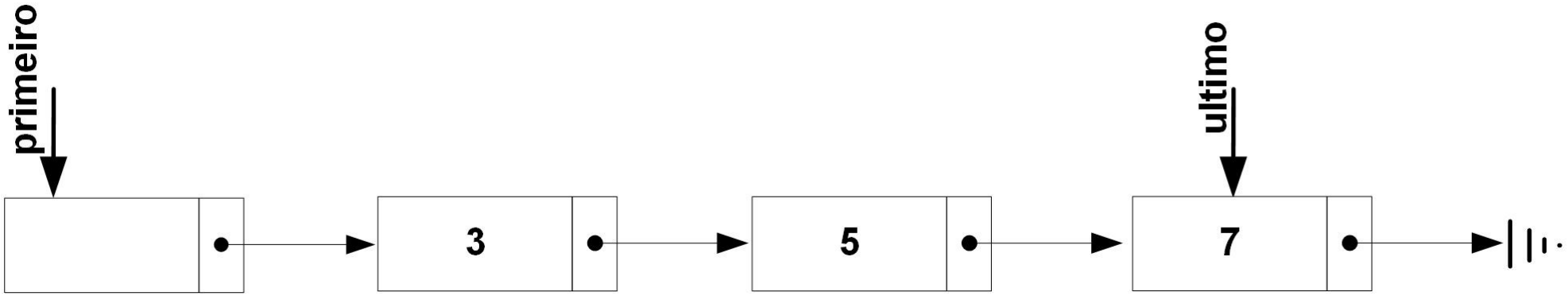
```
//inserirInicio(1)
public void inserirInicio(int x) {
    Celula tmp = new Celula(x);
    tmp.prox = primeiro.prox;
    primeiro.prox = tmp;
    if (primeiro == ultimo) {
       ultimo = tmp;
    }
    tmp = null;
}
```

Execute o método inserirInicio na figura abaixo!!!



# Remover no Fim

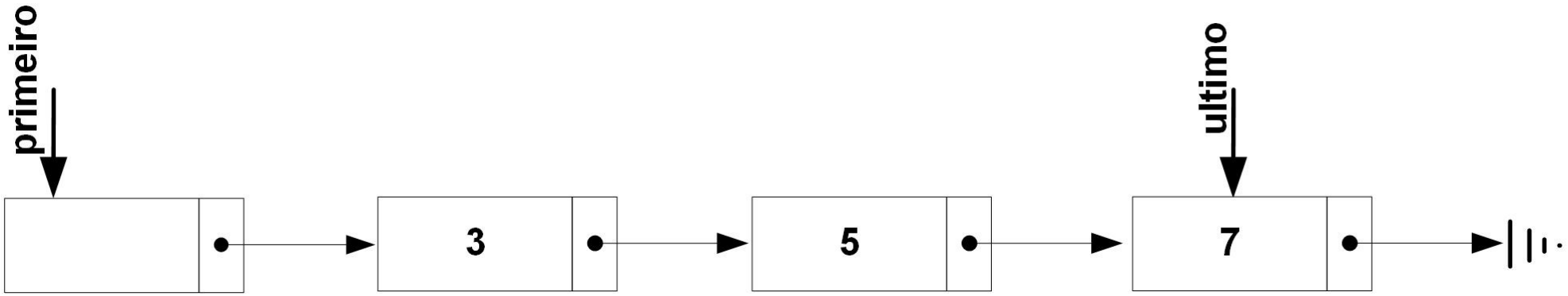
```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

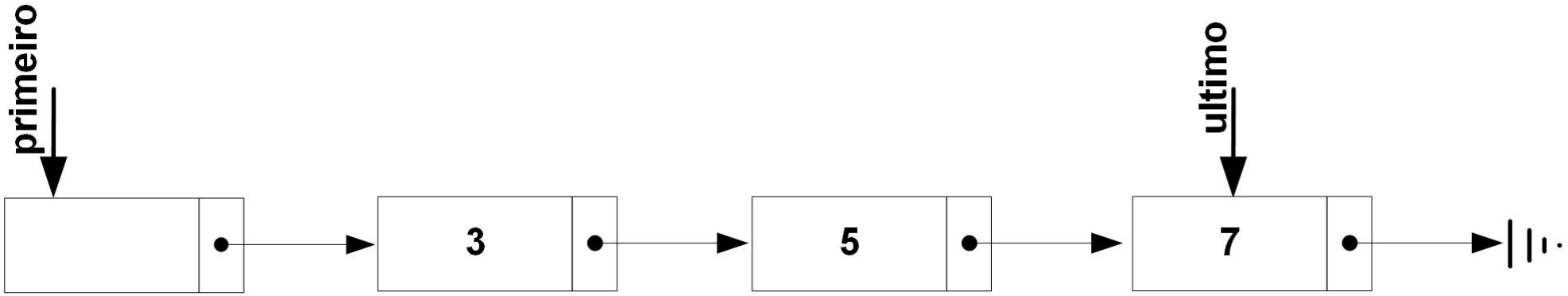


# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

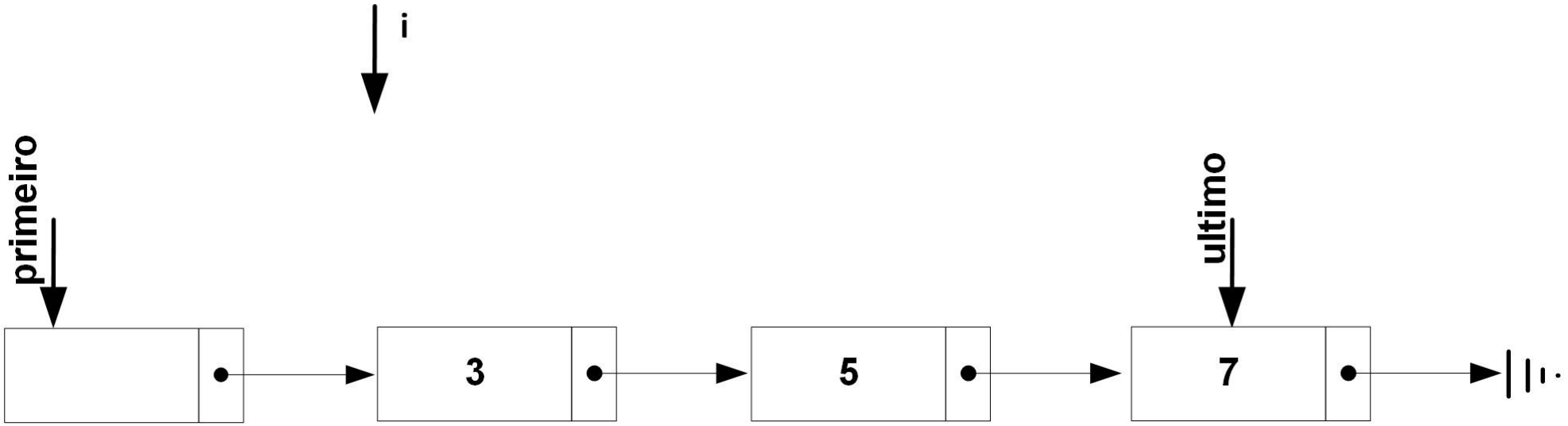
false



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

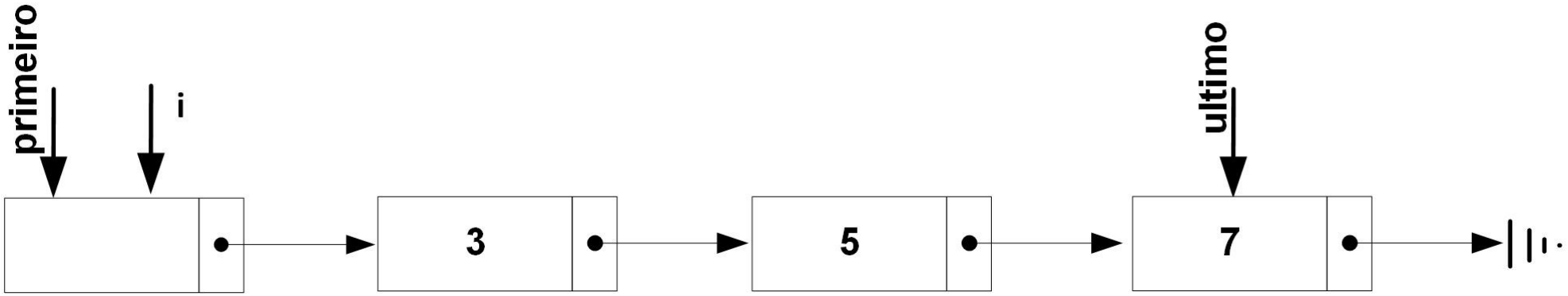
```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
        throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

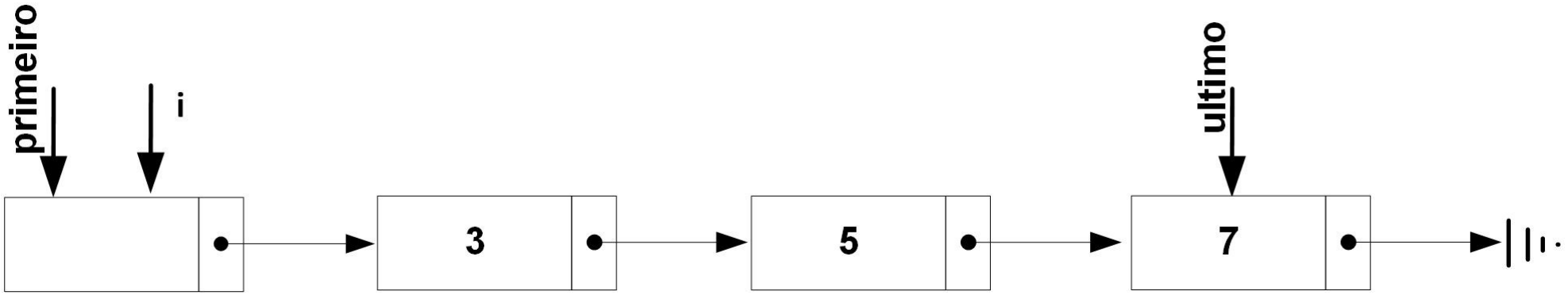


# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

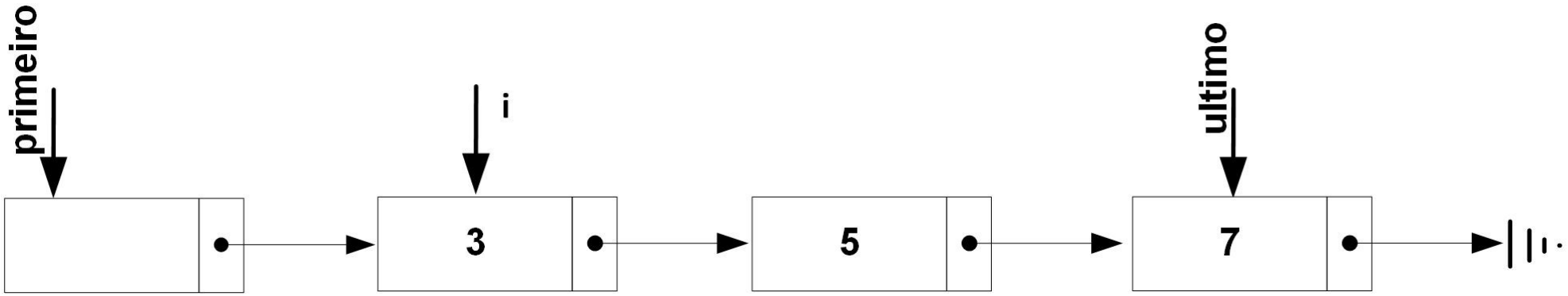
true



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

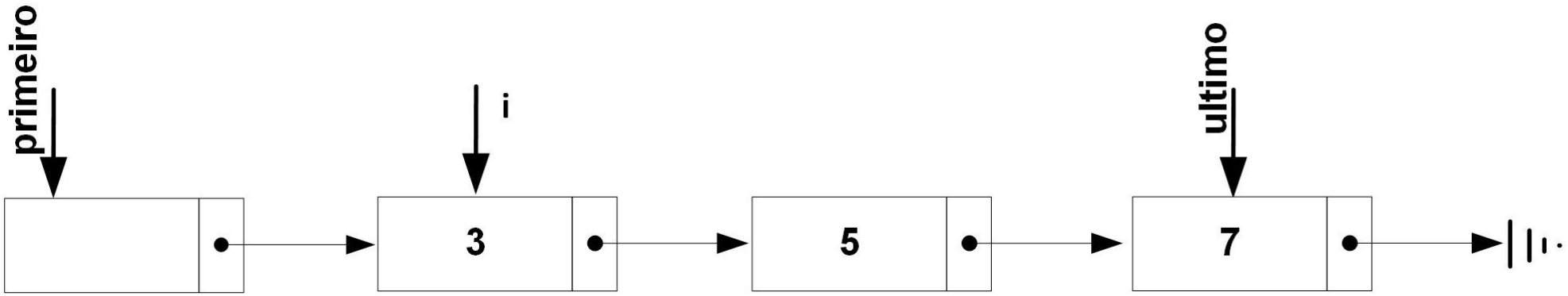


# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

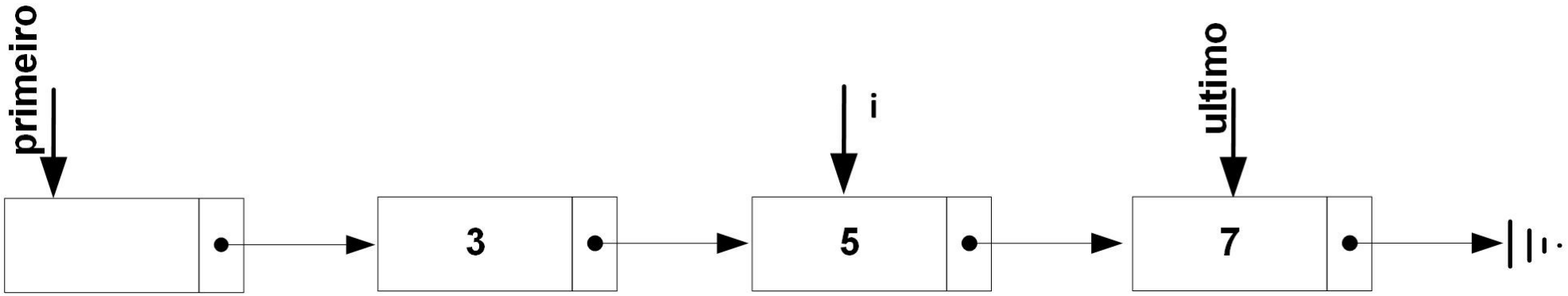
true



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

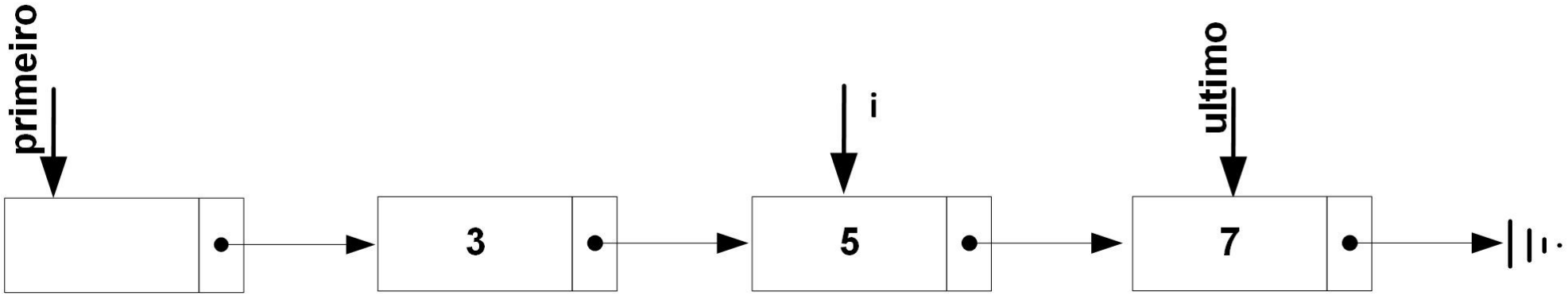


# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

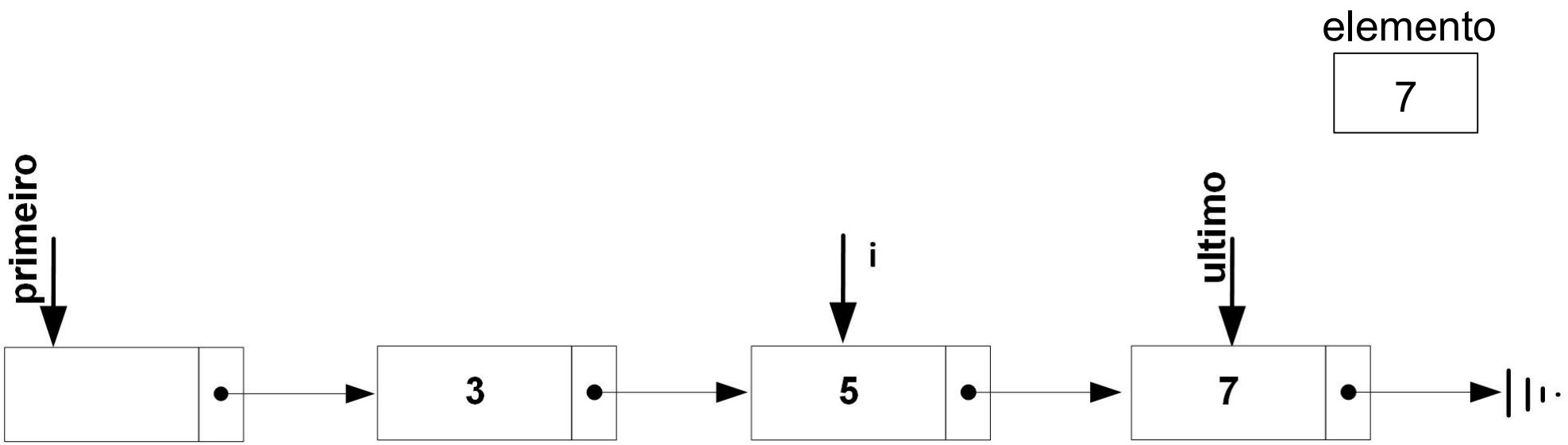
false



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

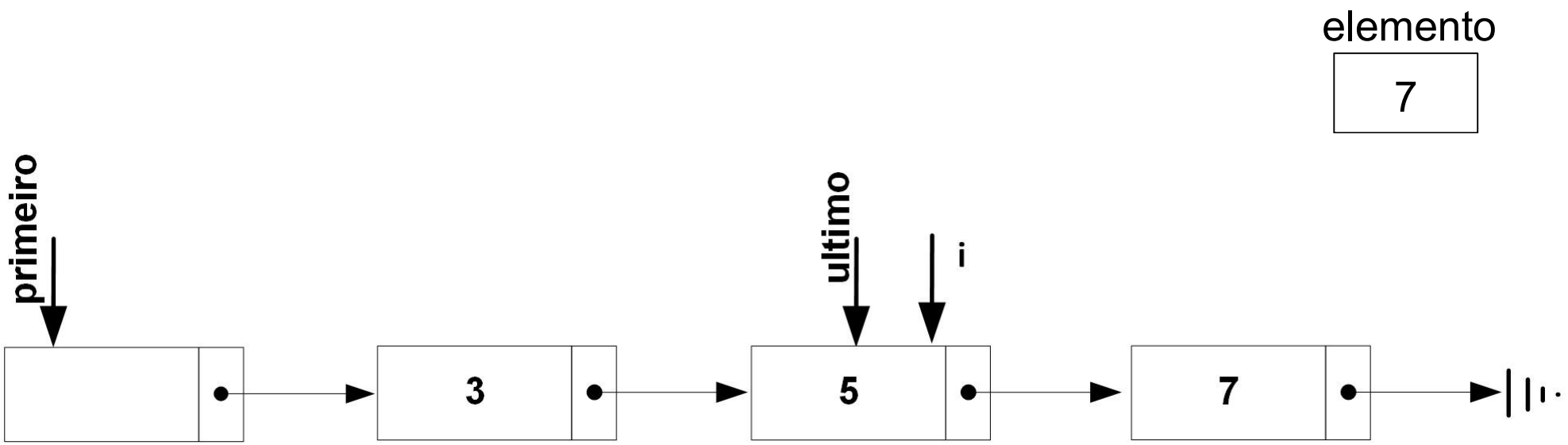
```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

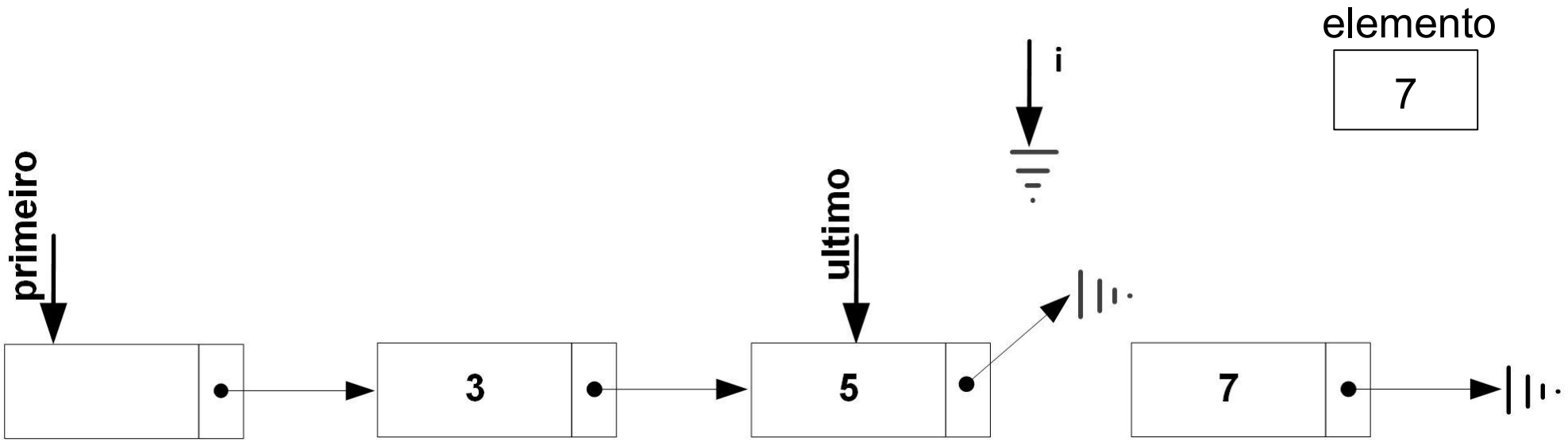
```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```



# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

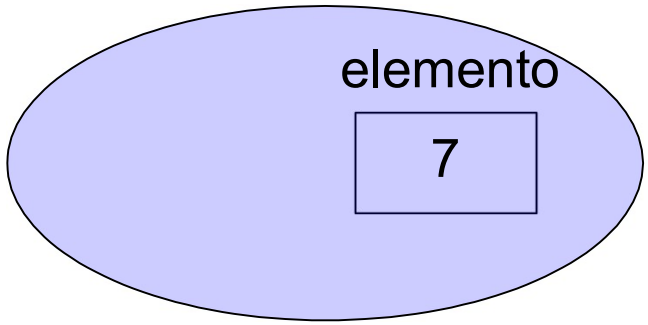
```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```

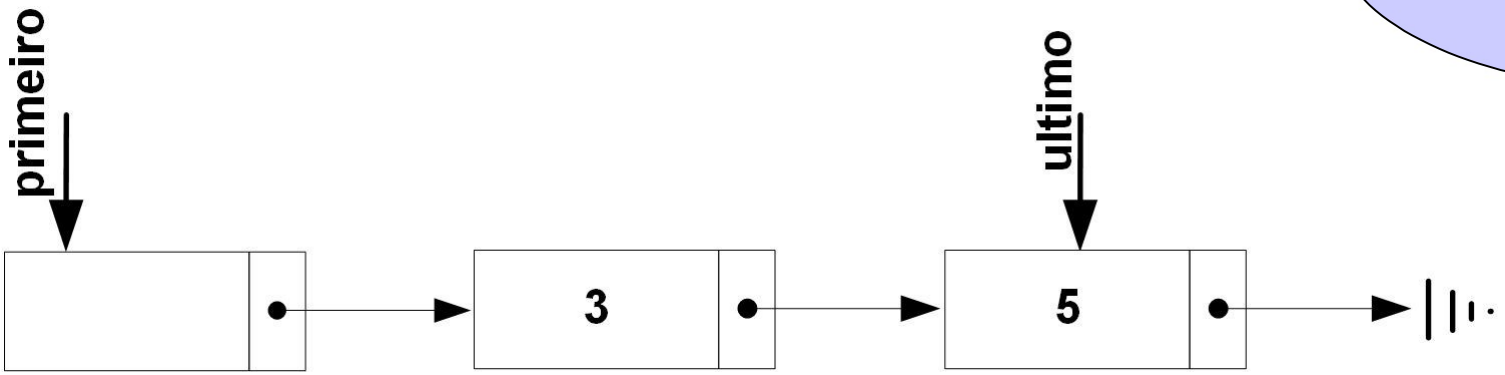


# Remover no Fim

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```

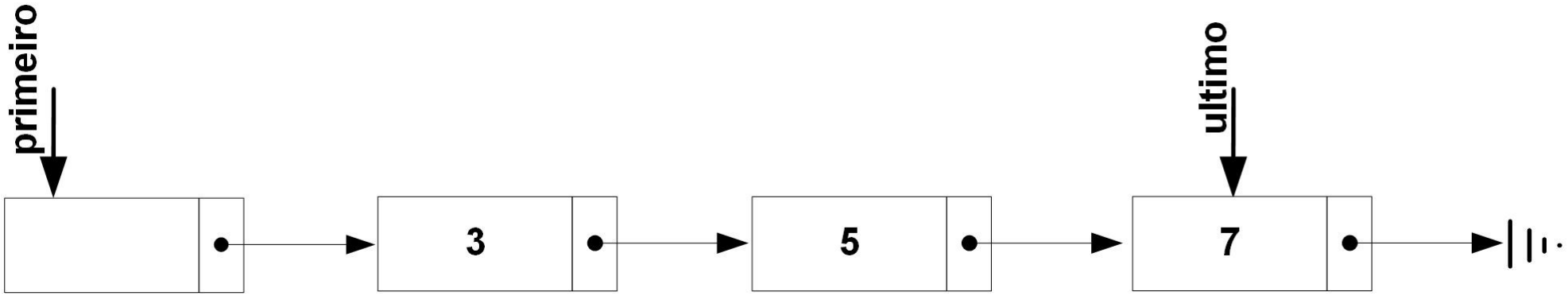
```
public int removerFim() throws Exception {
    if (primeiro == ultimo)
      throw new Exception("Erro!");
    Celula i;
    for(i = primeiro; i.prox != ultimo; i = i.prox);
    int elemento = ultimo.elemento;
    ultimo = i;
    i = ultimo.prox = null;
    return elemento;
}
```



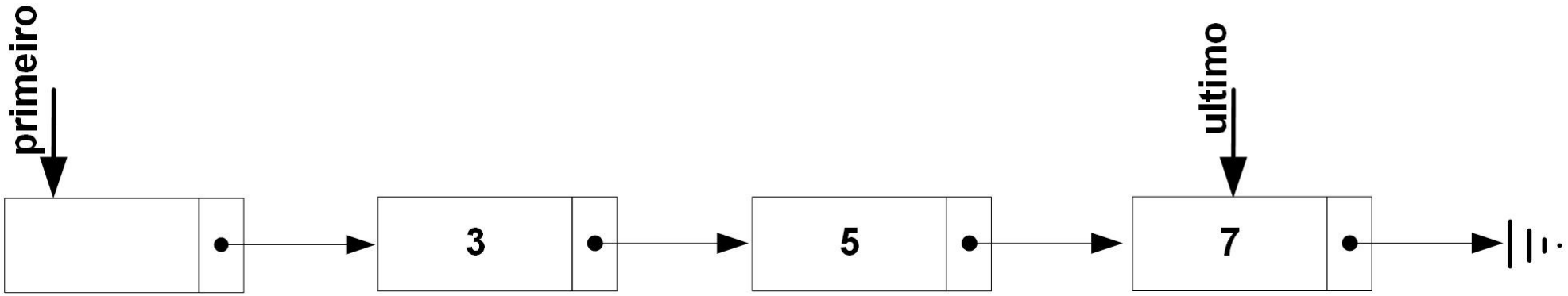
# Inserir

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```



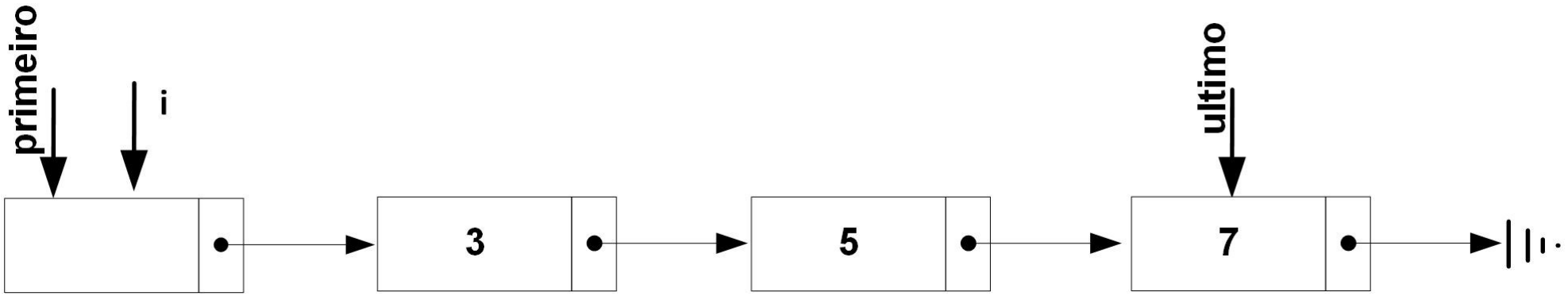
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



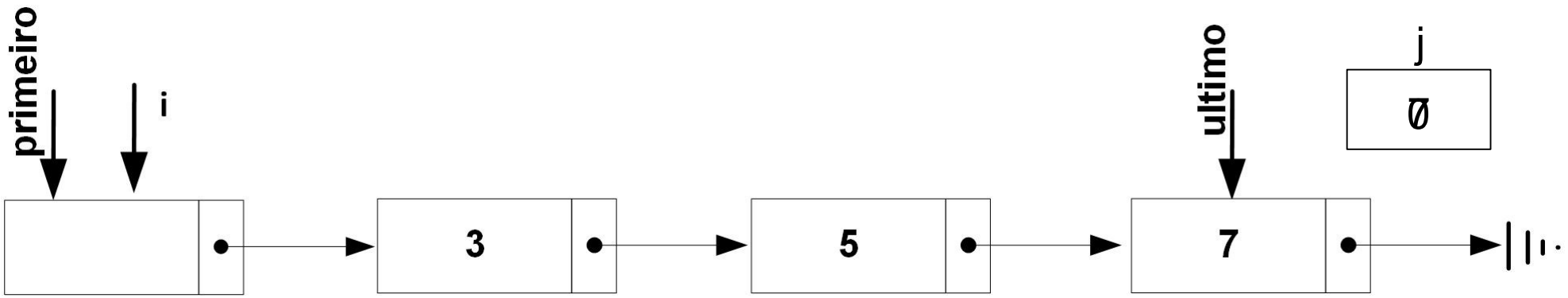
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){              inserirInicio(x);
   } else if (pos == tamanho){     inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



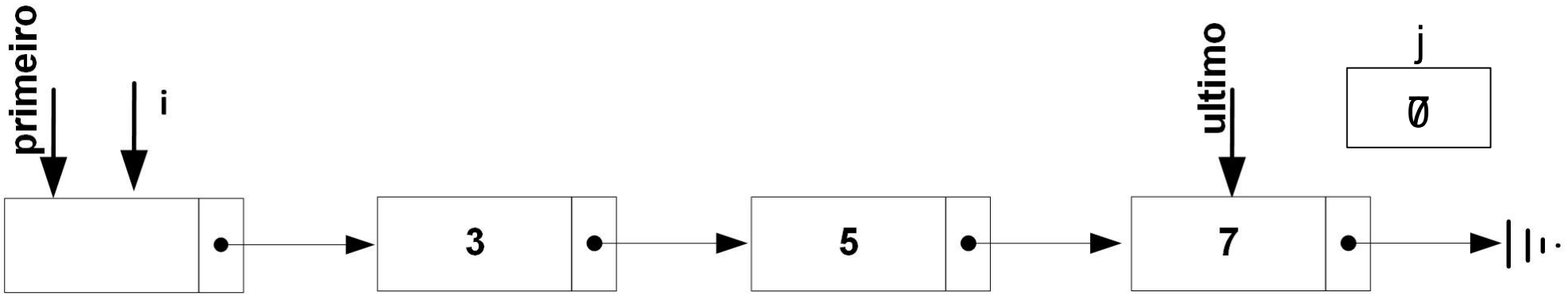
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



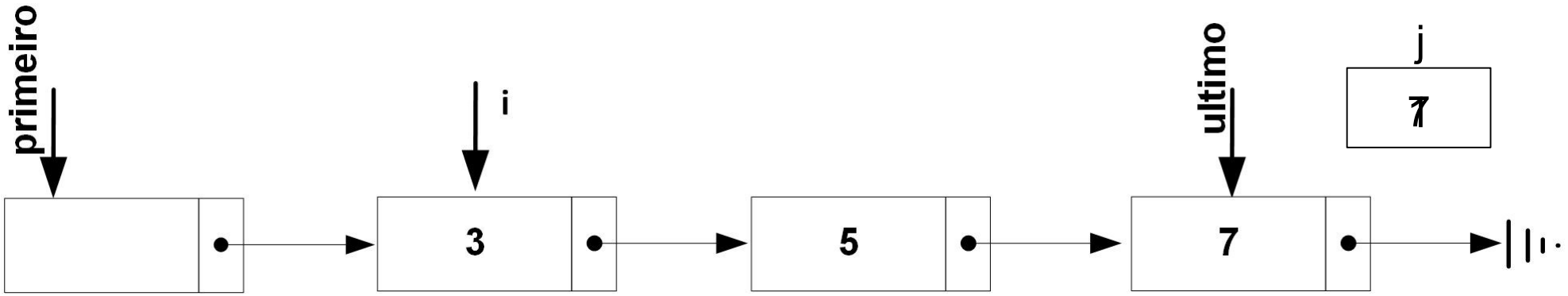
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;                                    true
        tmp = i = null;
} }
```



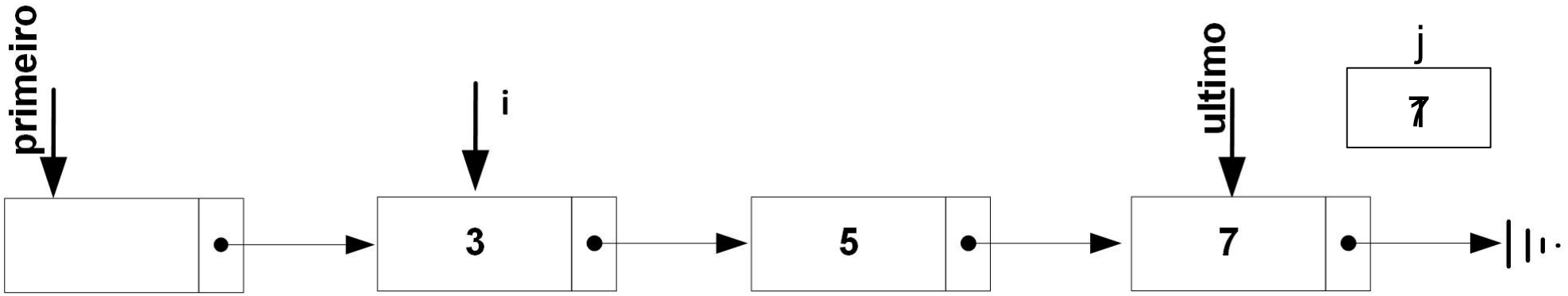
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



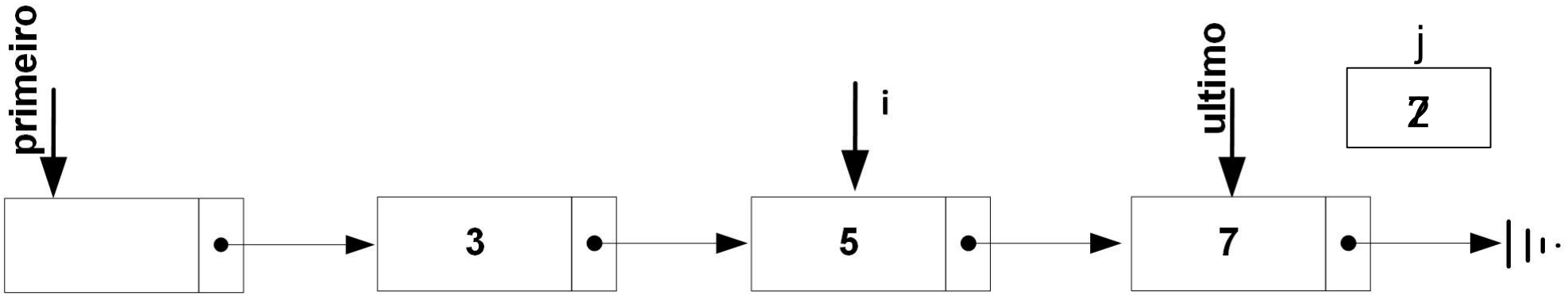
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;                                    true
        tmp = i = null;
} }
```



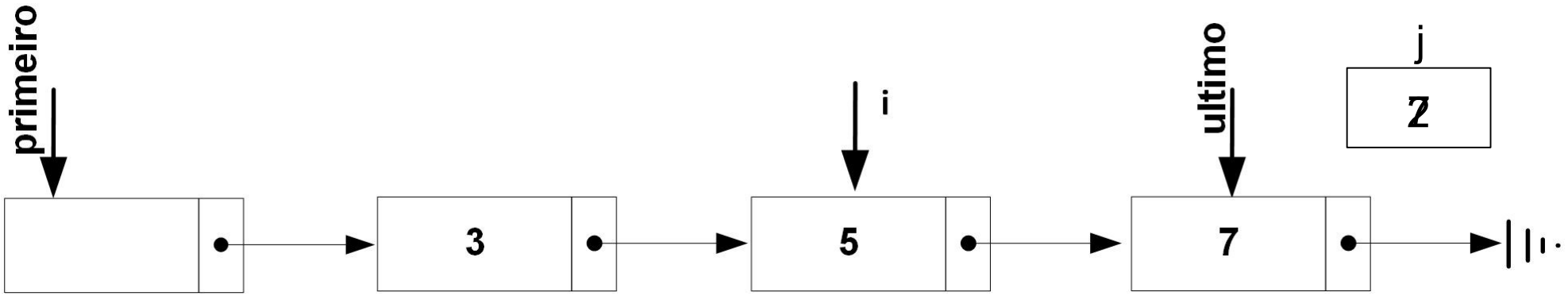
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



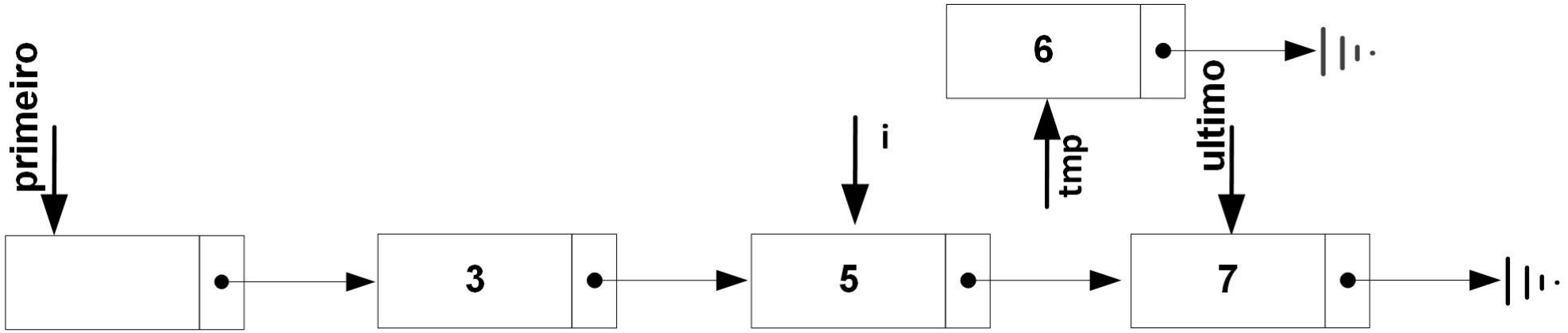
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;                                    false
        tmp = i = null;
} }
```



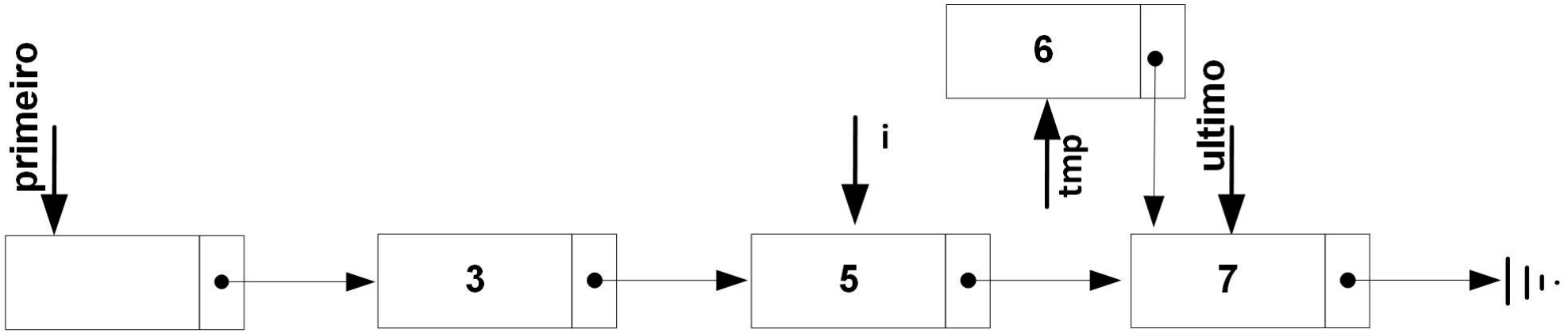
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



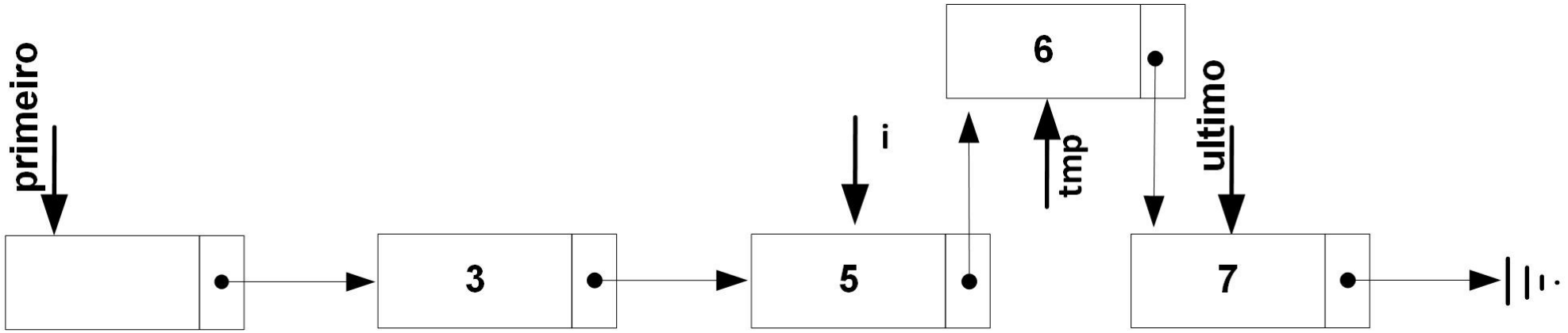
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



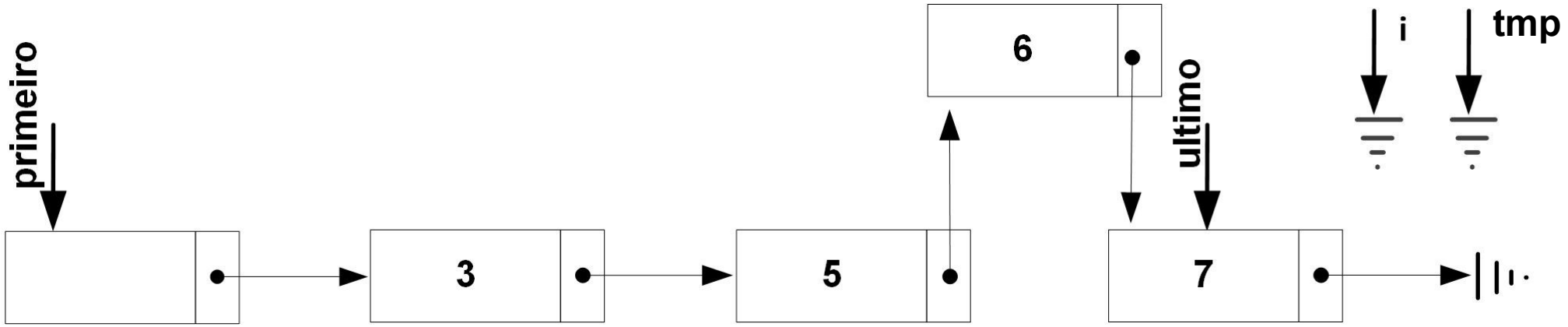
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



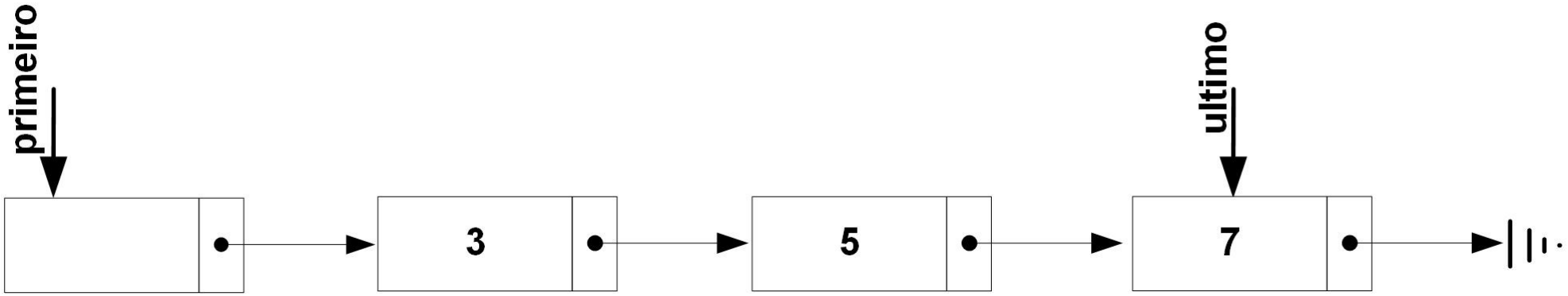
# Inserir

```
public void inserir(int x, int pos) throws Exception {        //Inserir(6, 2)
   int tamanho = tamanho();
   if (pos < 0 || pos > tamanho){   throw new Exception("Erro!");
   } else if (pos == 0){                inserirInicio(x);
   } else if (pos == tamanho){      inserirFim(x);
   } else {
        Celula i = primeiro;
        for(int j = 0; j < pos; j++, i = i.prox);
        Celula tmp = new Celula(x);
        tmp.prox = i.prox;
        i.prox = tmp;
        tmp = i = null;
} }
```



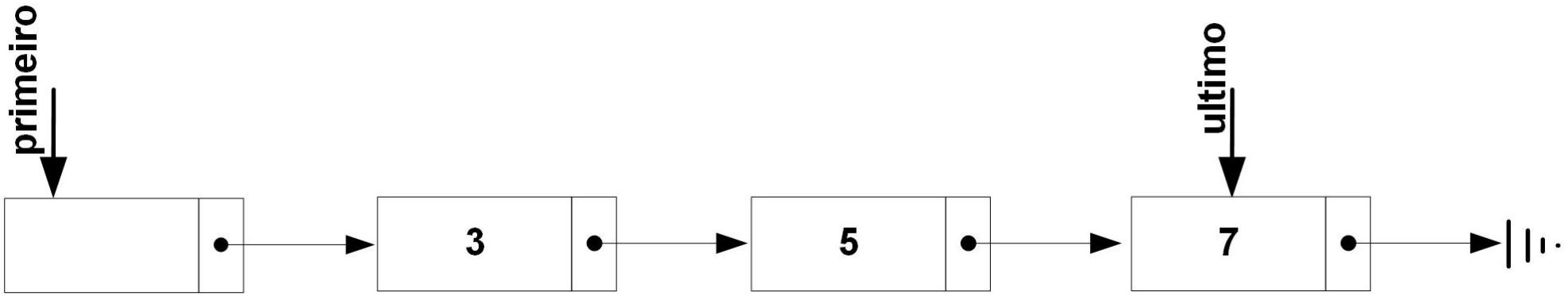
# Remover

```
class Lista {
    ...
    public void inserirInicio(int x) { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
}
```



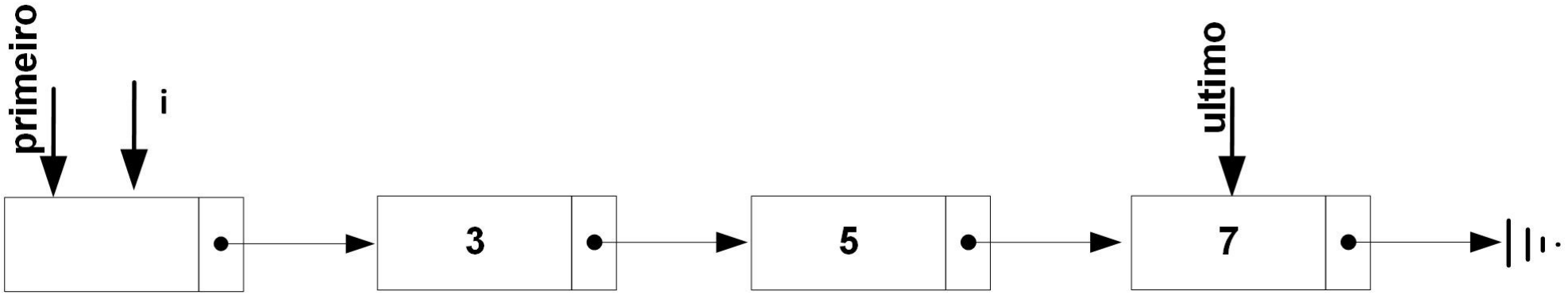
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```



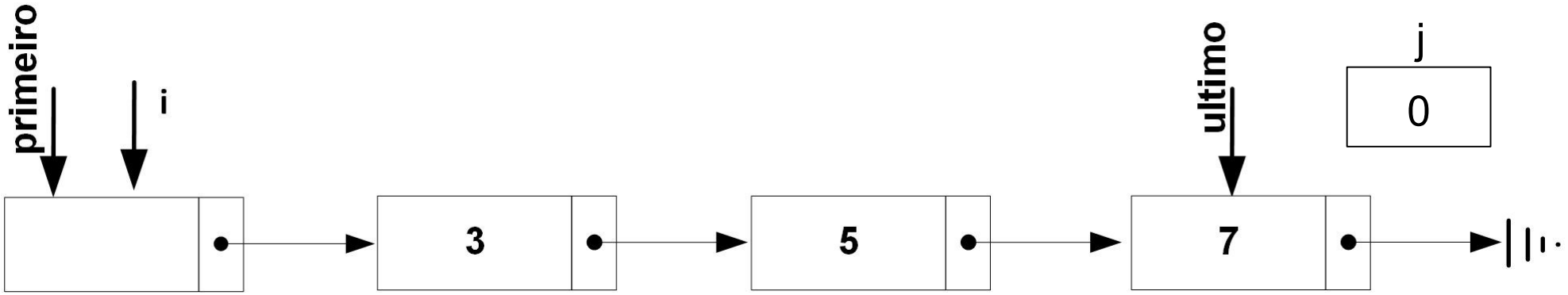
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```



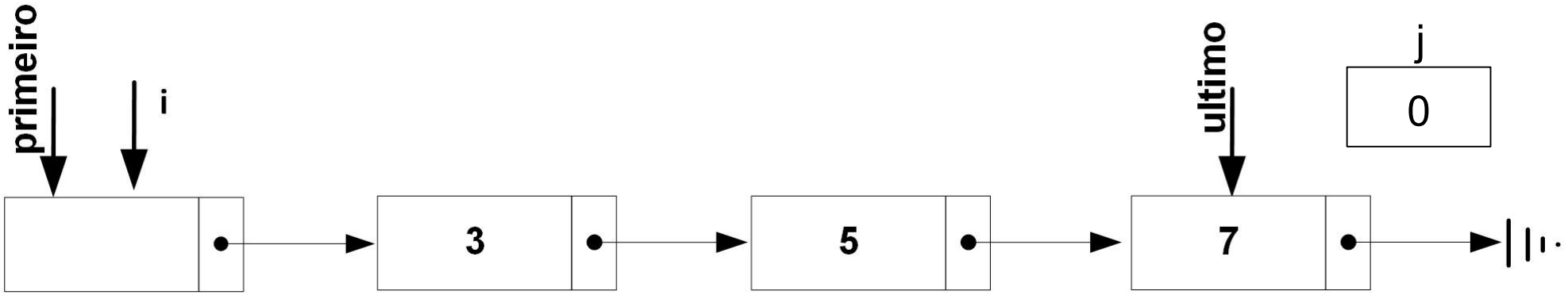
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```



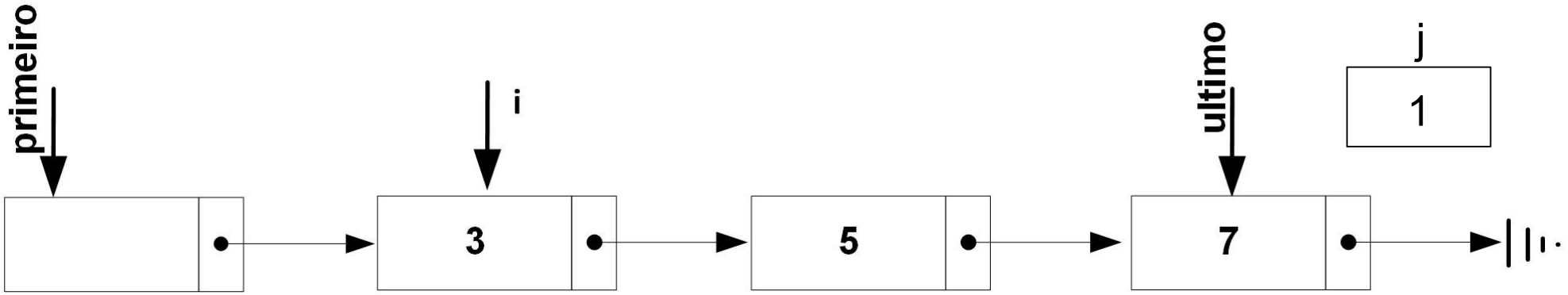
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }                                                          true
   return elemento;
}
```



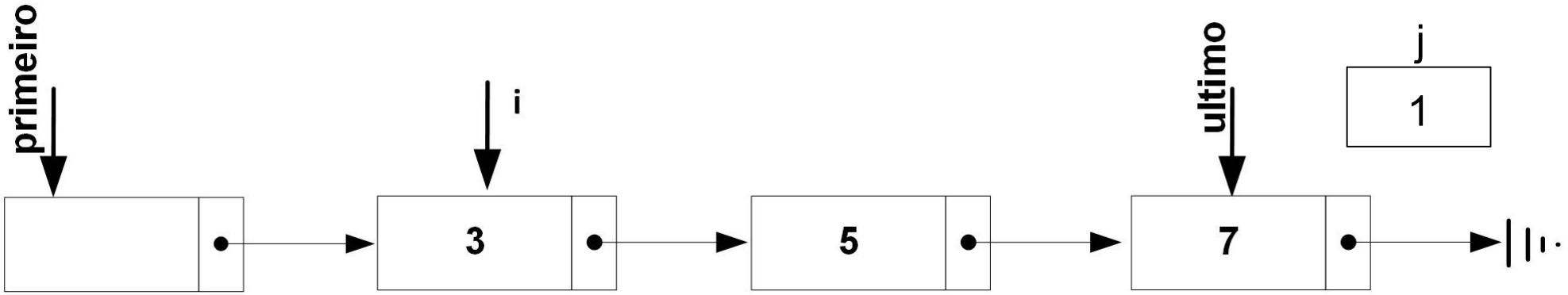
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```



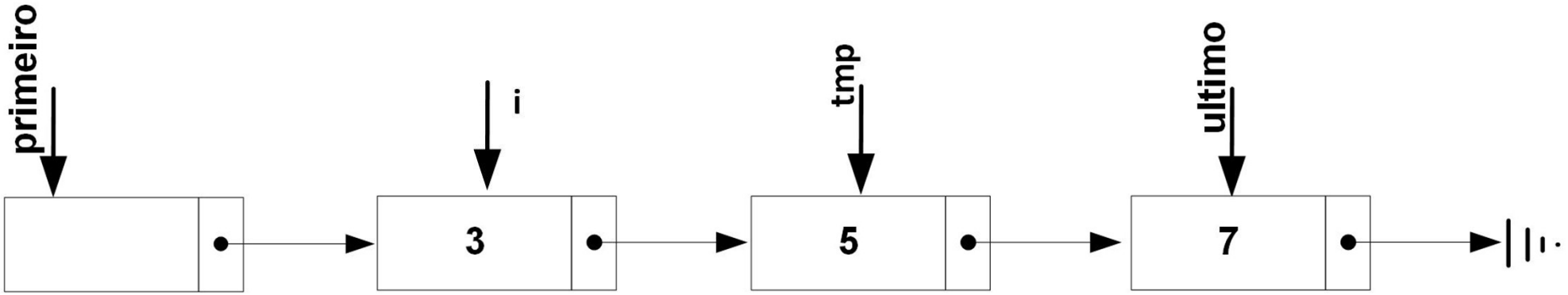
# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }                                                          false
   return elemento;
}
```



# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```

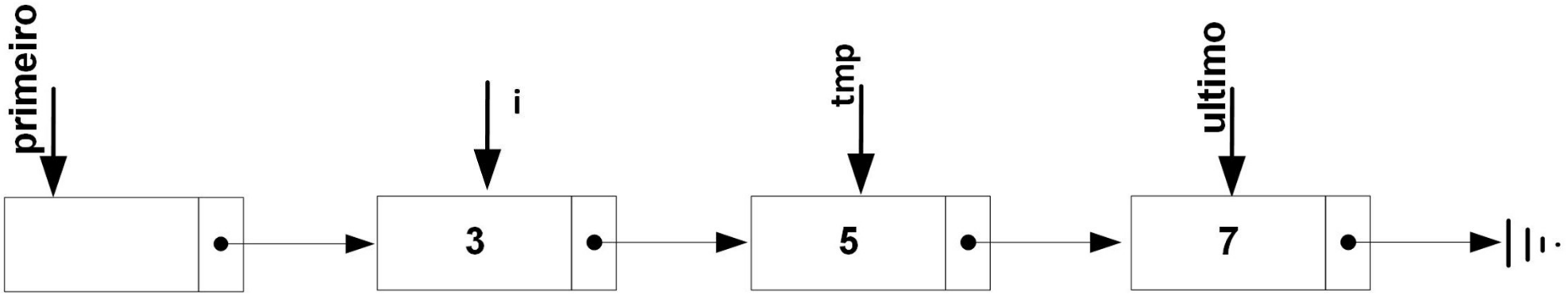


# Remover

```
public int remover(int pos ) throws Exception {          //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                  elemento = removerInicio();
   } else if (pos == tamanho - 1){         elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;          i.prox = tmp.prox;
         tmp.prox = null;                  i = tmp = null;
   }
   return elemento;
}
```

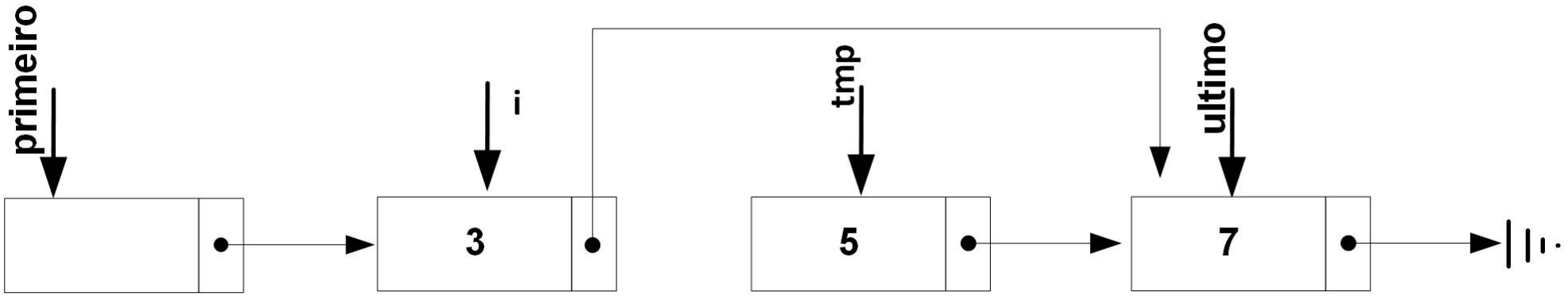
elemento

5



# Remover

```
public int remover(int pos ) throws Exception {          //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                  elemento = removerInicio();
   } else if (pos == tamanho - 1){         elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);                          elemento
         Celula tmp = i.prox;                                                  5
         elemento = tmp.elemento;      i.prox = tmp.prox;
         tmp.prox = null;              i = tmp = null;
   }
   return elemento;
}
```

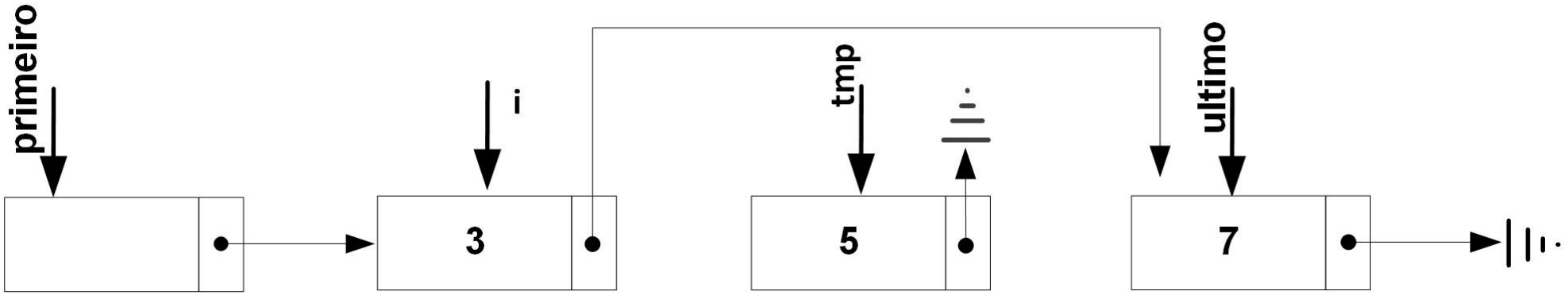


# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```

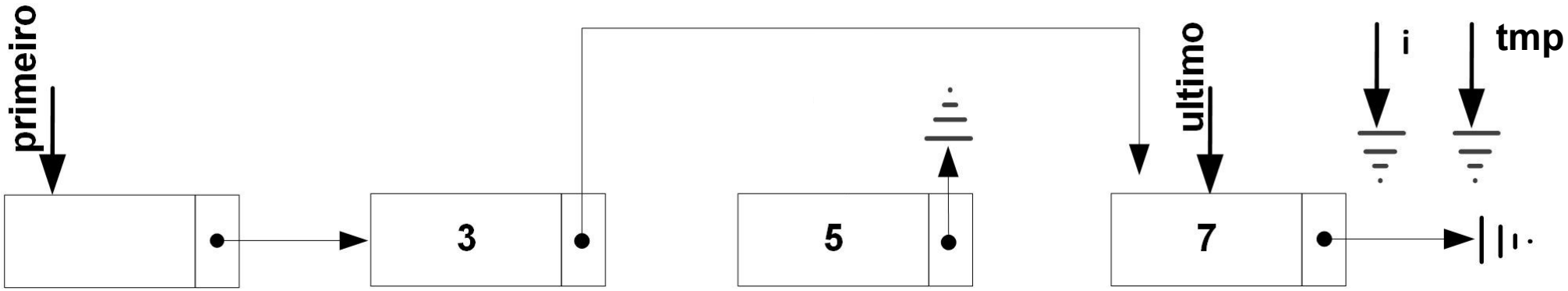
elemento

5



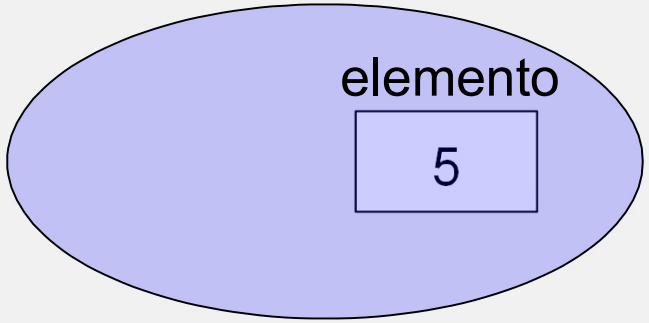
# Remover

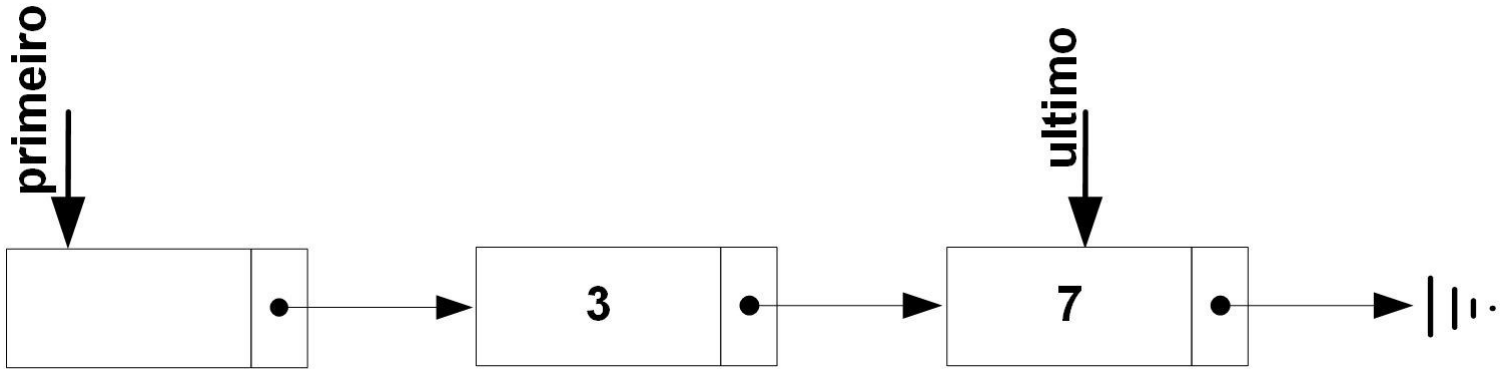
```
public int remover(int pos ) throws Exception {          //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                  elemento = removerInicio();
   } else if (pos == tamanho - 1){         elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);                          elemento
         Celula tmp = i.prox;                                                  5
         elemento = tmp.elemento;        i.prox = tmp.prox;
         tmp.prox = null;                i = tmp = null;
   }
   return elemento;
}
```



# Remover

```
public int remover(int pos ) throws Exception {              //remover(1)
   int elemento, tamanho = tamanho();
   if (primeiro == ultimo || pos < 0 || pos >= tamanho){ throw new Exception("Erro!");
   } else if (pos == 0) {                    elemento = removerInicio();
   } else if (pos == tamanho - 1){           elemento = removerFim();
   } else {
         Celula i = primeiro;
         for(int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;            i.prox = tmp.prox;
         tmp.prox = null;                    i = tmp = null;
   }
   return elemento;
}
```



# Classe Célula Dupla

```
class CelulaDupla {
    public int elemento;
    public CelulaDupla prox, ant;
    public CelulaDupla () {
        this(0);
    }
    public CelulaDupla (int x) {
        this.elemento = x;
        this.prox = this.ant = null;
    }
}
```

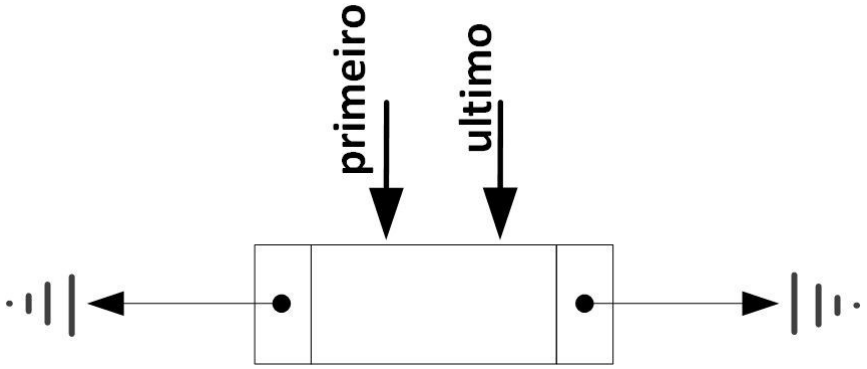
# Lista Dupla Flexível

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

Similar a Lista Simples, contudo, considerando o ponteiro ant

# Lista Dupla Flexível

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```



# Inserir no Início

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Inserir no Início

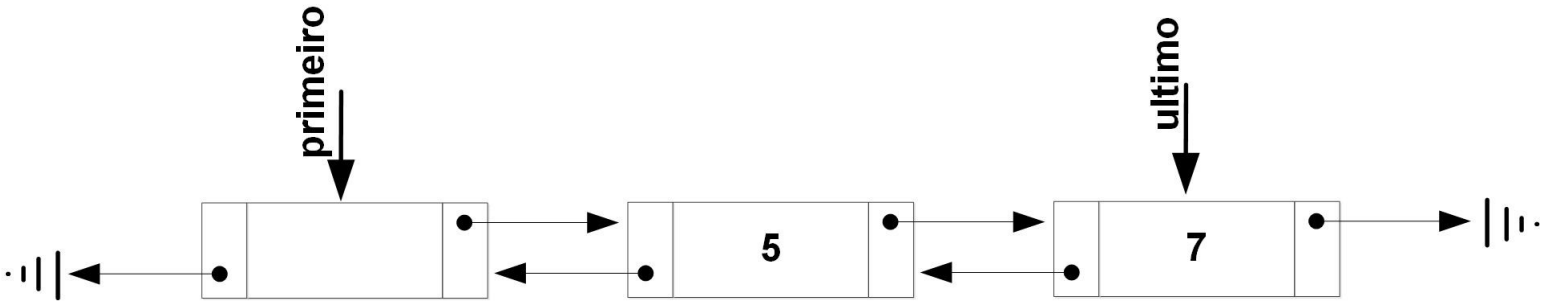
```
//LISTA DUPLA
public void inserirInicio(int x) {
      CelulaDupla tmp = new CelulaDupla(x);
      tmp.ant = primeiro;
      tmp.prox = primeiro.prox;
      primeiro.prox = tmp;
      if (primeiro == ultimo) {
            ultimo = tmp;
      } else {
            tmp.prox.ant = tmp;
      }
      tmp = null;
}
```

```
//LISTA SIMPLES
public void inserirInicio(int x) {
      Celula tmp = new Celula(x);

      tmp.prox = primeiro.prox;
      primeiro.prox = tmp;
      if (primeiro == ultimo) {
         ultimo = tmp;



      }
      tmp = null;
}
```

# Inserir no Início

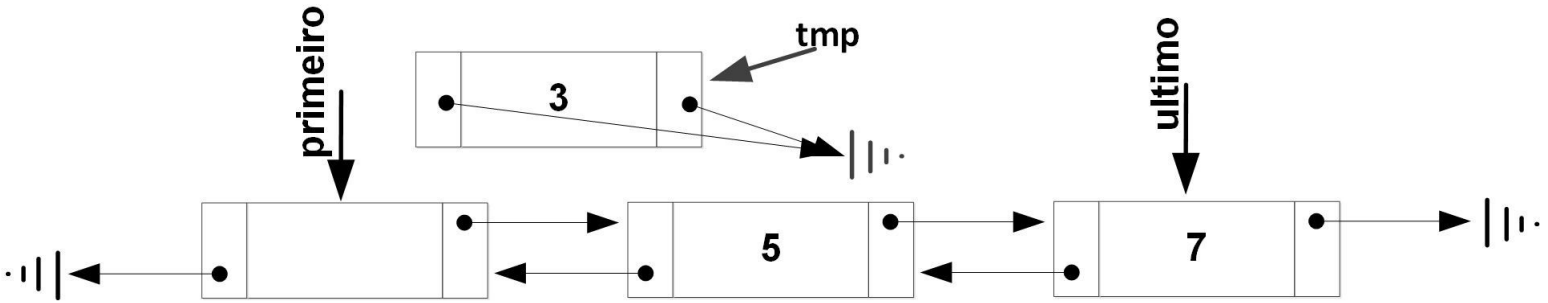
```
//LISTA DUPLA
public void inserirInicio(int x) {
        CelulaDupla tmp = new CelulaDupla(x);
        tmp.ant = primeiro;
        tmp.prox = primeiro.prox;
        primeiro.prox = tmp;
        if (primeiro == ultimo) {
                ultimo = tmp;
        } else {
                tmp.prox.ant = tmp;
        }
        tmp = null;
}
```

Supondo uma lista com os elementos 5 e 7, vamos inserir o 3 no início



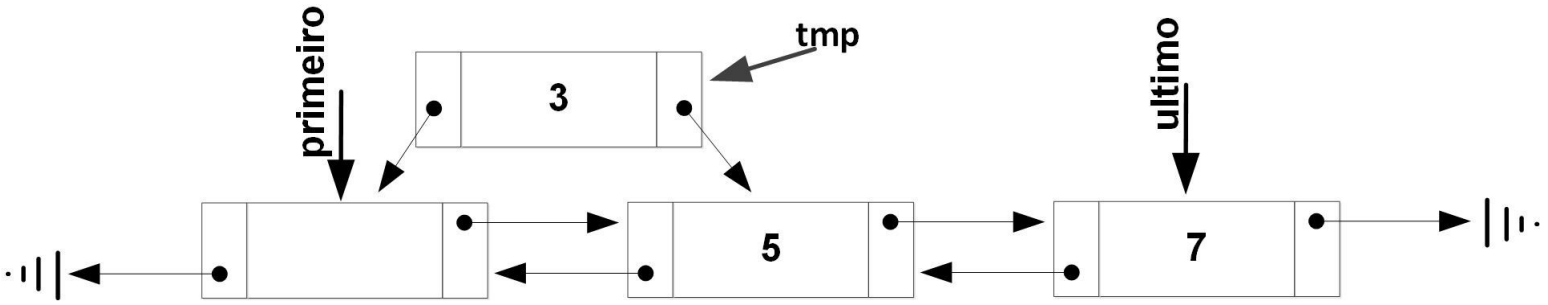
# Inserir no Início

```
//Inserindo o 3 no início
public void inserirInicio(int x) {
        CelulaDupla tmp = new CelulaDupla(x);
        tmp.ant = primeiro;
        tmp.prox = primeiro.prox;
        primeiro.prox = tmp;
        if (primeiro == ultimo) {
                ultimo = tmp;
        } else {
                tmp.prox.ant = tmp;
        }
        tmp = null;
}
```



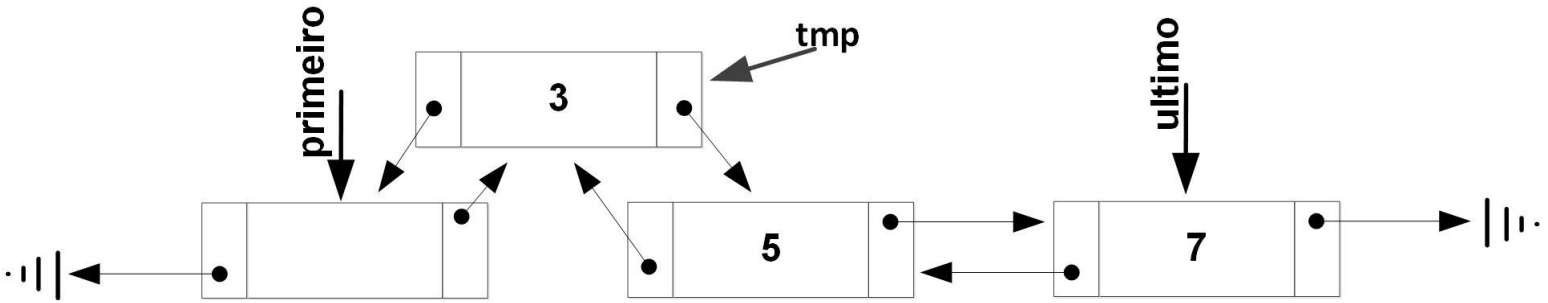
# Inserir no Início

```
//Inserindo o 3 no início
public void inserirInicio(int x) {
        CelulaDupla tmp = new CelulaDupla(x);
        tmp.ant = primeiro;
        tmp.prox = primeiro.prox;
        primeiro.prox = tmp;
        if (primeiro == ultimo) {
                ultimo = tmp;
        } else {
                tmp.prox.ant = tmp;
        }
        tmp = null;
}
```



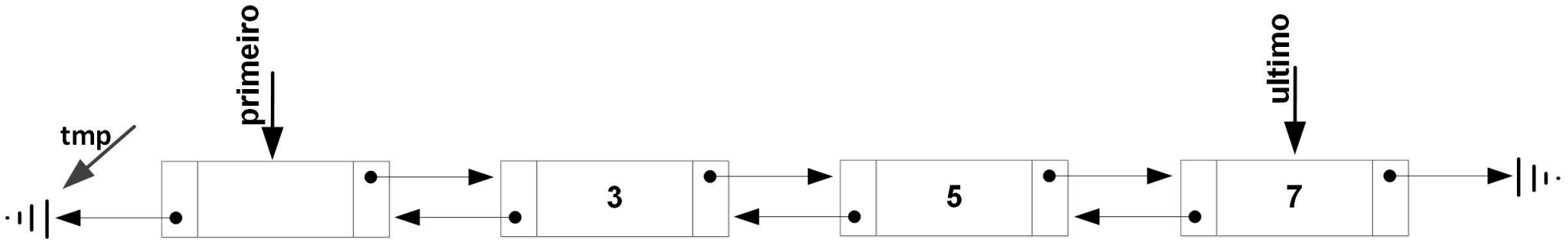
# Inserir no Início

```
//Inserindo o 3 no início
public void inserirInicio(int x) {
        CelulaDupla tmp = new CelulaDupla(x);
        tmp.ant = primeiro;
        tmp.prox = primeiro.prox;
        primeiro.prox = tmp;
        if (primeiro == ultimo) {
                ultimo = tmp;
        } else {
                tmp.prox.ant = tmp;
        }
        tmp = null;
}
```



# Inserir no Início

```
//Inserindo o 3 no início
public void inserirInicio(int x) {
        CelulaDupla tmp = new CelulaDupla(x);
        tmp.ant = primeiro;
        tmp.prox = primeiro.prox;
        primeiro.prox = tmp;
        if (primeiro == ultimo) {
                ultimo = tmp;
        } else {
                tmp.prox.ant = tmp;
        }
        tmp = null;
}
```



# Inserir no Fim

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Inserir no Fim

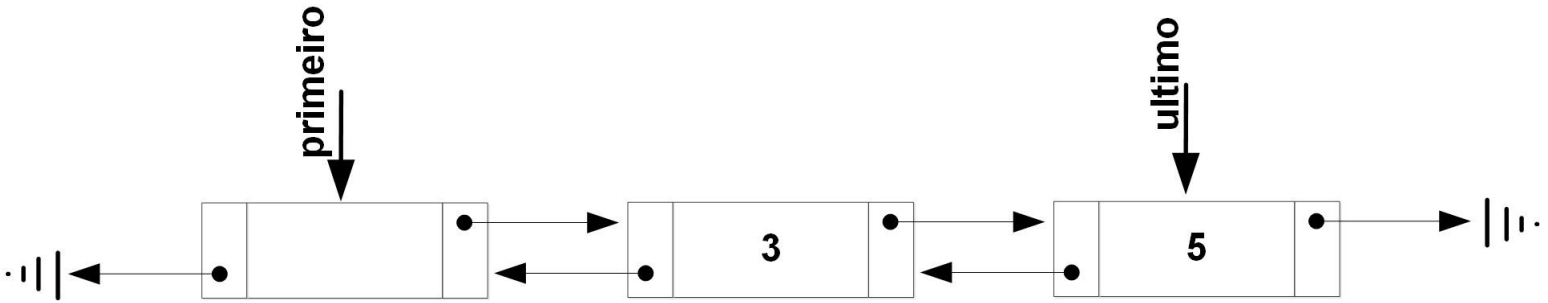
```
//LISTA DUPLA
public void inserirFim(int x) {
      ultimo.prox = new CelulaDupla(x);
      ultimo.prox.ant = ultimo;
      ultimo = ultimo.prox;
}
```

```
//LISTA SIMPLES
public void inserirFIm(int x) {
      ultimo.prox = new Celula(x);

      ultimo = ultimo.prox;
}
```

# Inserir no Fim

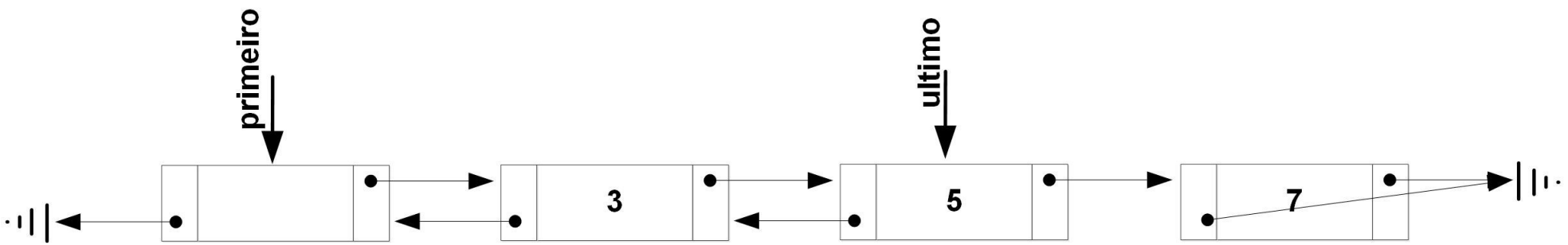
```
//LISTA DUPLA
public void inserirFim(int x) {
        ultimo.prox = new CelulaDupla(x);
        ultimo.prox.ant = ultimo;
        ultimo = ultimo.prox;
}
```

Supondo uma lista com os elementos 3 e 5, vamos inserir o 7 no fim



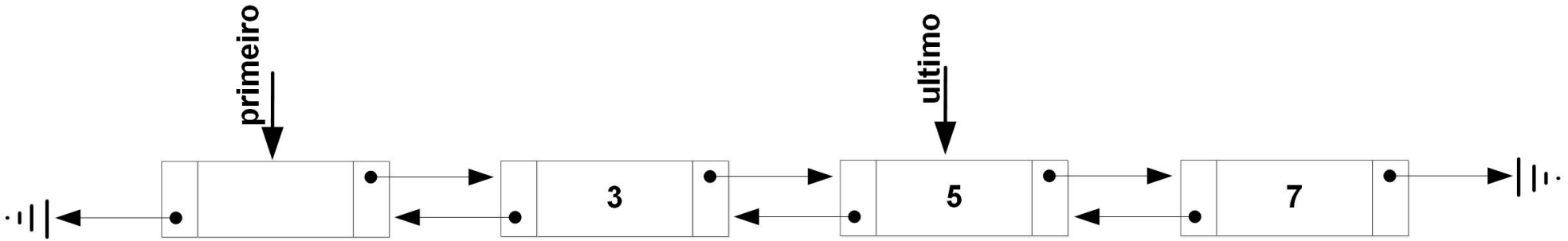
# Inserir no Fim

```
//LISTA DUPLA
public void inserirFim(int x) {
        ultimo.prox = new CelulaDupla(x);
        ultimo.prox.ant = ultimo;
        ultimo = ultimo.prox;
}
```



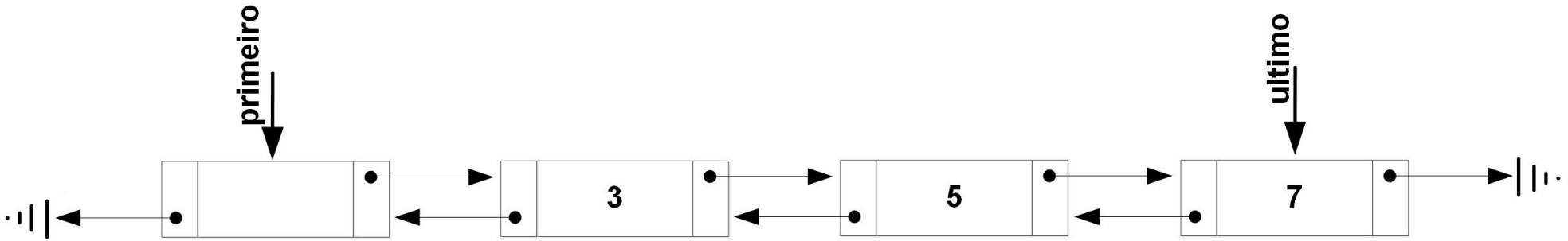
# Inserir no Fim

```
//LISTA DUPLA
public void inserirFim(int x) {
      ultimo.prox = new CelulaDupla(x);
      ultimo.prox.ant = ultimo;
      ultimo = ultimo.prox;
}
```



# Inserir no Fim

```
//LISTA DUPLA
public void inserirFim(int x) {
        ultimo.prox = new CelulaDupla(x);
        ultimo.prox.ant = ultimo;
        ultimo = ultimo.prox;
}
```

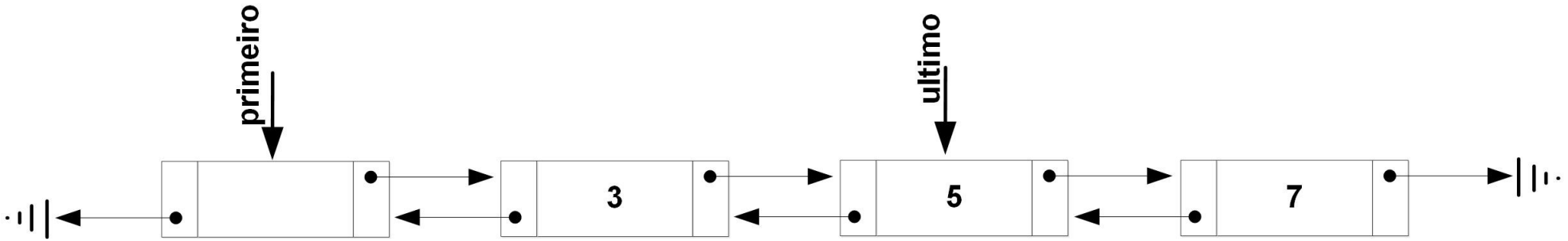


# Exercício

```
//LISTA DUPLA
public void inserirFim(int x) {
        ultimo.prox = new CelulaDupla(x);
        ultimo.prox.ant = ultimo;
        ultimo = ultimo.prox;
}
```

- A linha marcada pode ser substituída pelo código abaixo?

*ultimo = ultimo.prox.ant.prox.ant.prox*



# Remover no Início

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Remover no Início

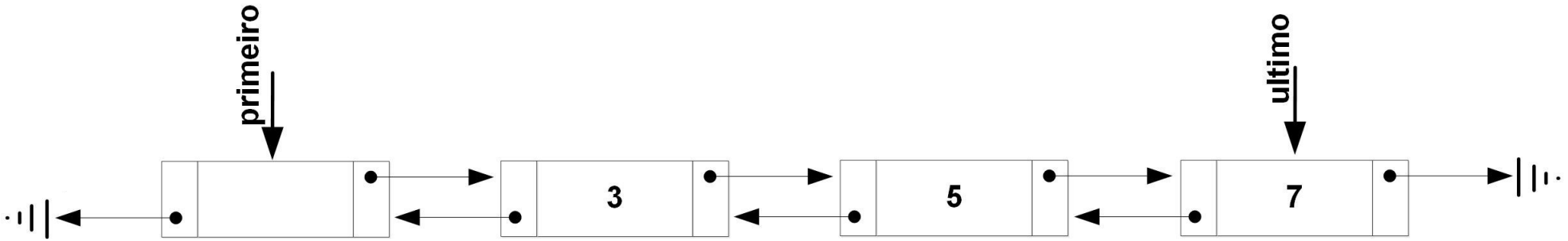
```
//LISTA DUPLA
public int removerInicio() throws Exception {
        if (primeiro == ultimo)
                throw new Exception("Erro!");
        CelulaDupla tmp = primeiro;
        primeiro = primeiro.prox;
        int elemento = primeiro.elemento;
        tmp.prox = primeiro.ant = null;
        tmp = null;
        return elemento;
}
```

```
//LISTA SIMPLES
public int removerInicio() throws Exception {
        if (primeiro == ultimo)
                throw new Exception("Erro!");
        Celula tmp = primeiro;
        primeiro = primeiro.prox;
        int elemento = primeiro.elemento;
        tmp.prox = null;
        tmp = null;
        return elemento;
}
```

# Remover no Início

```
//LISTA DUPLA
public int removerInicio() throws Exception {
        if (primeiro == ultimo)
                throw new Exception("Erro!");
        CelulaDupla tmp = primeiro;
        primeiro = primeiro.prox;
        int elemento = primeiro.elemento;
        tmp.prox = primeiro.ant = null;
        tmp = null;
        return elemento;
}
```

**Exercício**: Supondo uma lista com os elementos 3, 5 e 7, execute o remover no início



# Remover no Fim

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Remover no Fim

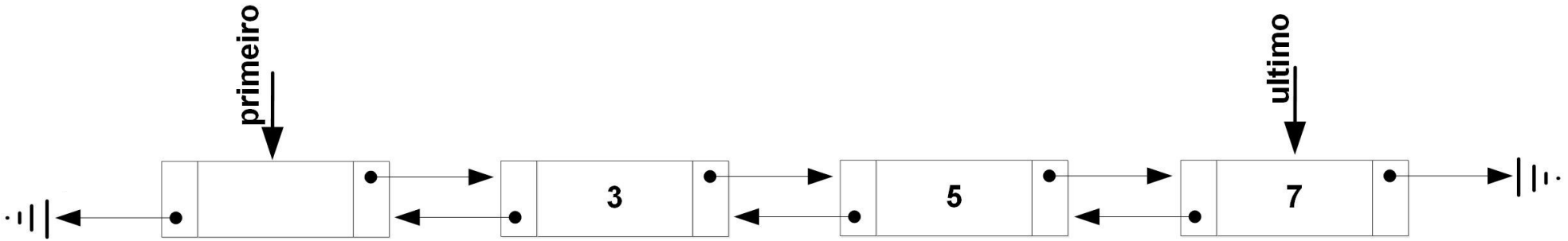
```
//LISTA DUPLA
public int removerFim() throws Exception {
      if (primeiro == ultimo)
        throw new Exception("Erro!");


       int elemento = ultimo.elemento;
       ultimo = ultimo.ant;
       ultimo.prox.ant = null;
       ultimo.prox = null;
       return elemento;
}
```

```
//LISTA SIMPLES
public int removerFim() throws Exception {
      if (primeiro == ultimo)
        throw new Exception("Erro!");
      Celula i;
      for(i = primeiro; i.prox != ultimo; i = i.prox);
      int elemento = ultimo.elemento;
      ultimo = i;

      i = ultimo.prox = null;
      return elemento;
}
```

# Remover no Fim

```
//LISTA DUPLA
public int removerFim() throws Exception {
      if (primeiro == ultimo)
        throw new Exception("Erro!");


       int elemento = ultimo.elemento;
       ultimo = ultimo.ant;
       ultimo.prox.ant = null;
       ultimo.prox = null;
       return elemento;
}
```

**Exercício**: Supondo uma lista com os elementos 3, 5 e 7, execute o remover no fim



# Inserir

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Inserir

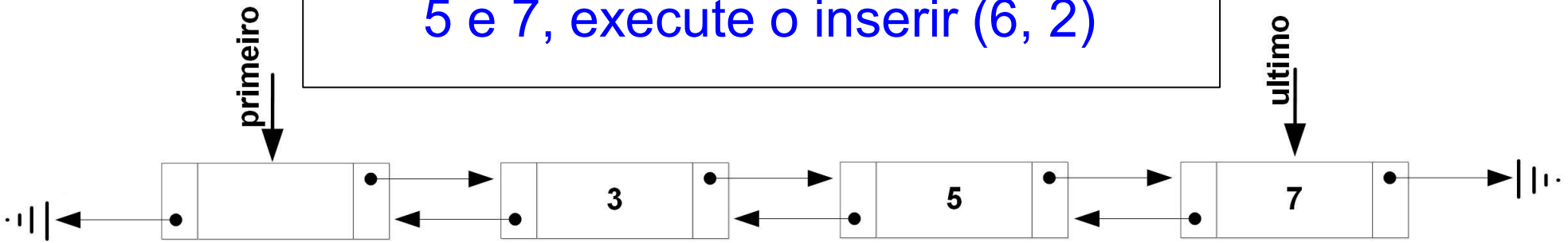
```
//LISTA DUPLA
public void inserir(int x, int pos) throws
Exception {
      int tamanho = tamanho();
      if (pos < 0 || pos > tamanho){
         throw new Exception("Erro!");
      } else if (pos == 0){ inserirInicio(x);
      } else if (pos == tamanho){ inserirFim(x);
      } else {
         CelulaDupla i = primeiro;
         for (int j = 0; j < pos; j++, i = i.prox);

         CelulaDupla tmp = new CelulaDupla(x);
         tmp.ant = i;
         tmp.prox = i.prox;
         tmp.ant.prox = tmp.prox.ant = tmp;
         tmp = i = null;
      }
   }
```

```
//LISTA SIMPLES
public void inserir(int x, int pos) throws
Exception {
      int tamanho = tamanho();
      if (pos < 0 || pos > tamanho){
         throw new Exception("Erro!");
      } else if (pos == 0){    inserirInicio(x);
      } else if (pos == tamanho){ inserirFim(x);
      } else {
         Celula i = primeiro;
         for (int j = 0; j < pos; j++, i = i.prox);

         Celula tmp = new Celula(x);

         tmp.prox = i.prox;
         i.prox = tmp;
         tmp = i = null;
      }
   }
```

# Inserir

```
//LISTA DUPLA
public void inserir(int x, int pos) throws Exception {
      int tamanho = tamanho();
      if (pos < 0 || pos > tamanho){                    throw new Exception("Erro!");
      } else if (pos == 0){                             inserirInicio(x);
      } else if (pos == tamanho){                   inserirFim(x);
      } else {
         CelulaDupla i = primeiro;                      for (int j = 0; j < pos; j++, i = i.prox);

         CelulaDupla tmp = new CelulaDupla(x);
         tmp.ant = i;                                   tmp.prox = i.prox;
         tmp.ant.prox = tmp.prox.ant = tmp;         tmp = i = null;
      }
   }
```

**Exercício**: Supondo a lista com o 3, 5 e 7, execute o inserir (6, 2)



# Remover

```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

# Remover

```
//LISTA DUPLA
public int remover(int pos) throws Exception {
      int elemento, tamanho = tamanho();
      if (primeiro == ultimo){
         throw new Exception("Erro!");
      } else if (pos < 0 || pos >= tamanho){
         throw new Exception("Erro!");
      } else if (pos == 0){
         elemento = removerInicio();
      } else if (pos == tamanho - 1){
         elemento = removerFim();
      } else {
         CelulaDupla i = primeiro.prox;
         for (int j = 0; j < pos; j++, i = i.prox);
         i.ant.prox = i.prox;
         i.prox.ant = i.ant;
         elemento = i.elemento;
         i.prox = i.ant = null;
         i = null;
      }
      return elemento;
   }
```

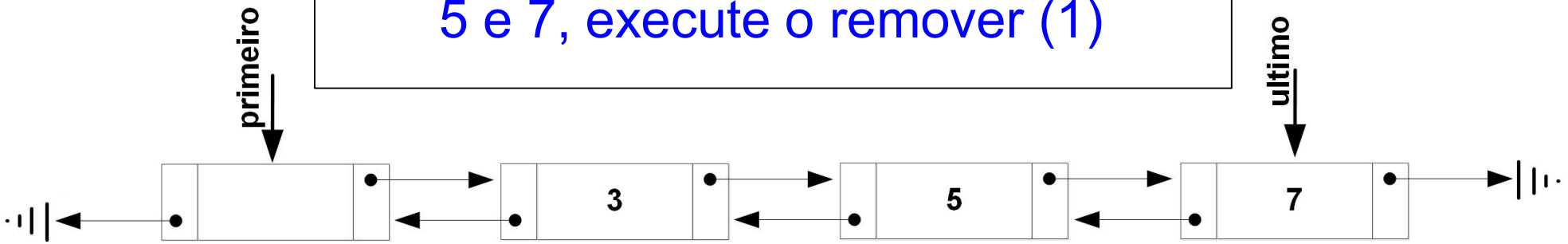
```
//LISTA SIMPLES
public int remover(int pos) throws Exception {
      int elemento, tamanho = tamanho();
      if (primeiro == ultimo){
         throw new Exception("Erro!");
      } else if (pos < 0 || pos >= tamanho){
         throw new Exception("Erro!");
      } else if (pos == 0){
         elemento = removerInicio();
      } else if (pos == tamanho - 1){
         elemento = removerFim();
      } else {
         Celula i = primeiro;
         for (int j = 0; j < pos; j++, i = i.prox);
         Celula tmp = i.prox;
         elemento = tmp.elemento;
         i.prox = tmp.prox;
         tmp.prox = null;
         i = tmp = null;
      }
      return elemento;
   }
```

# Remover

```
//LISTA DUPLA
public int remover(int pos) throws Exception {
      int elemento, tamanho = tamanho();
      if (primeiro == ultimo){
      } else if (pos < 0 || pos >= tamanho){
      } else if (pos == 0){
      } else if (pos == tamanho - 1){
      } else {
         CelulaDupla i = primeiro;
         i.ant.prox = i.prox;
         elemento = i.elemento;
      }
      return elemento;
   }
```

```
throw new Exception("Erro!");
throw new Exception("Erro!");
elemento = removerInicio();
elemento = removerFim();

for (int j = 0; j <= pos; j++, i = i.prox);
i.prox.ant = i.ant;
i = i.prox = i.ant = null;
```

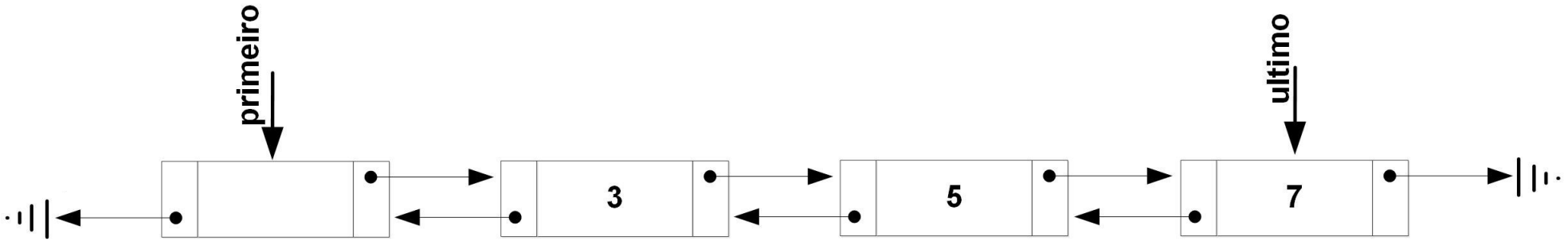
**Exercício**: Supondo a lista com o 3, 5 e 7, execute o remover (1)



# Mostrar

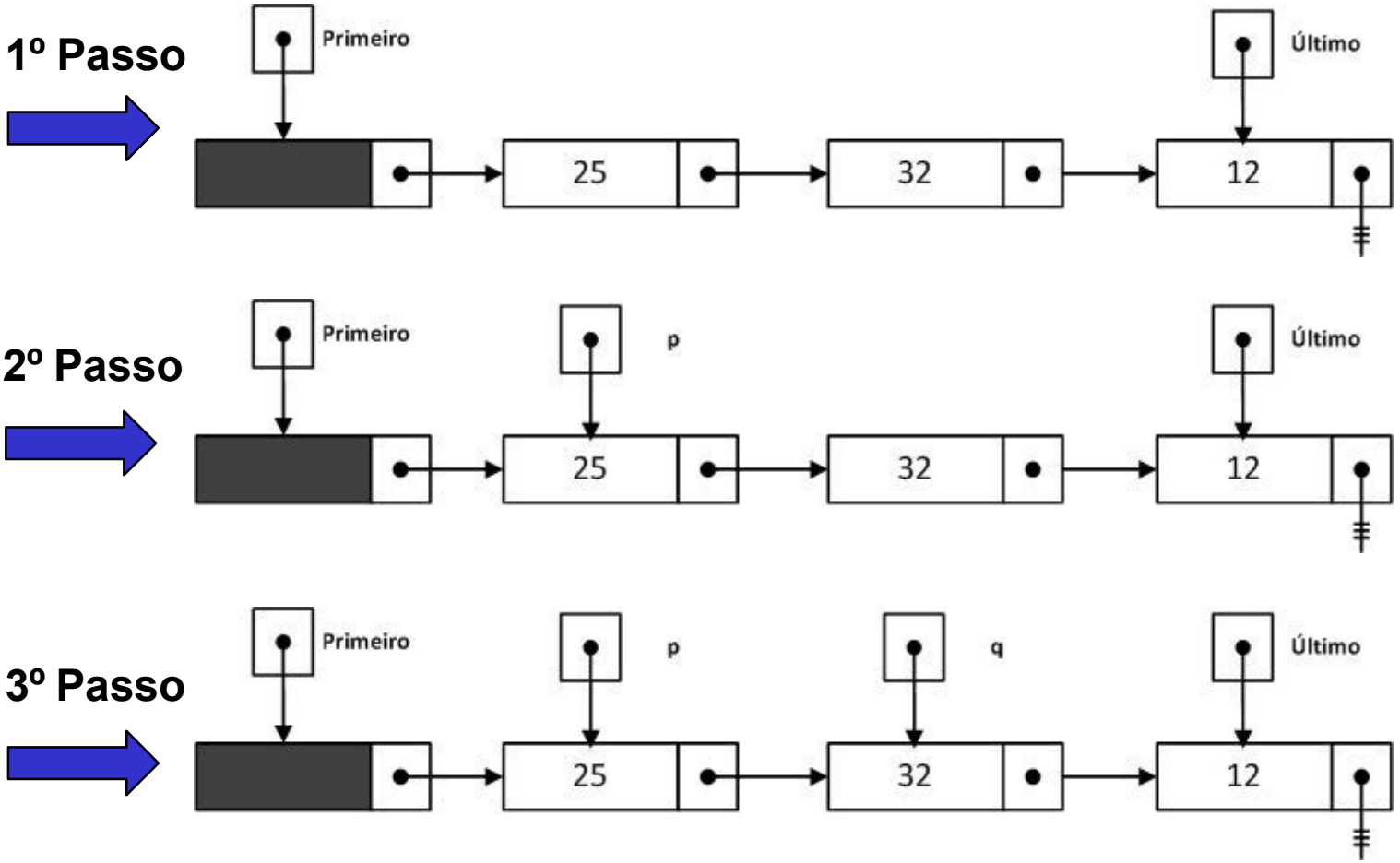
```
class ListaDupla {
    private CelulaDupla primeiro, ultimo;
    public ListaDupla () {
        primeiro = new CelulaDupla();
        ultimo = primeiro;
    }
    public void inserirInicio(int x) { ... }
    public void inserirFim(int x) { ... }
    public int removerInicio() { ... }
    public int removerFim() { ... }
    public void inserir(int x, int pos) { ... }
    public int remover(int pos) { ... }
    public void mostrar() { ... }
}
```

**Exercício**: Implemente o mostrar e o execute para uma lista com os elementos 3, 5 e 7



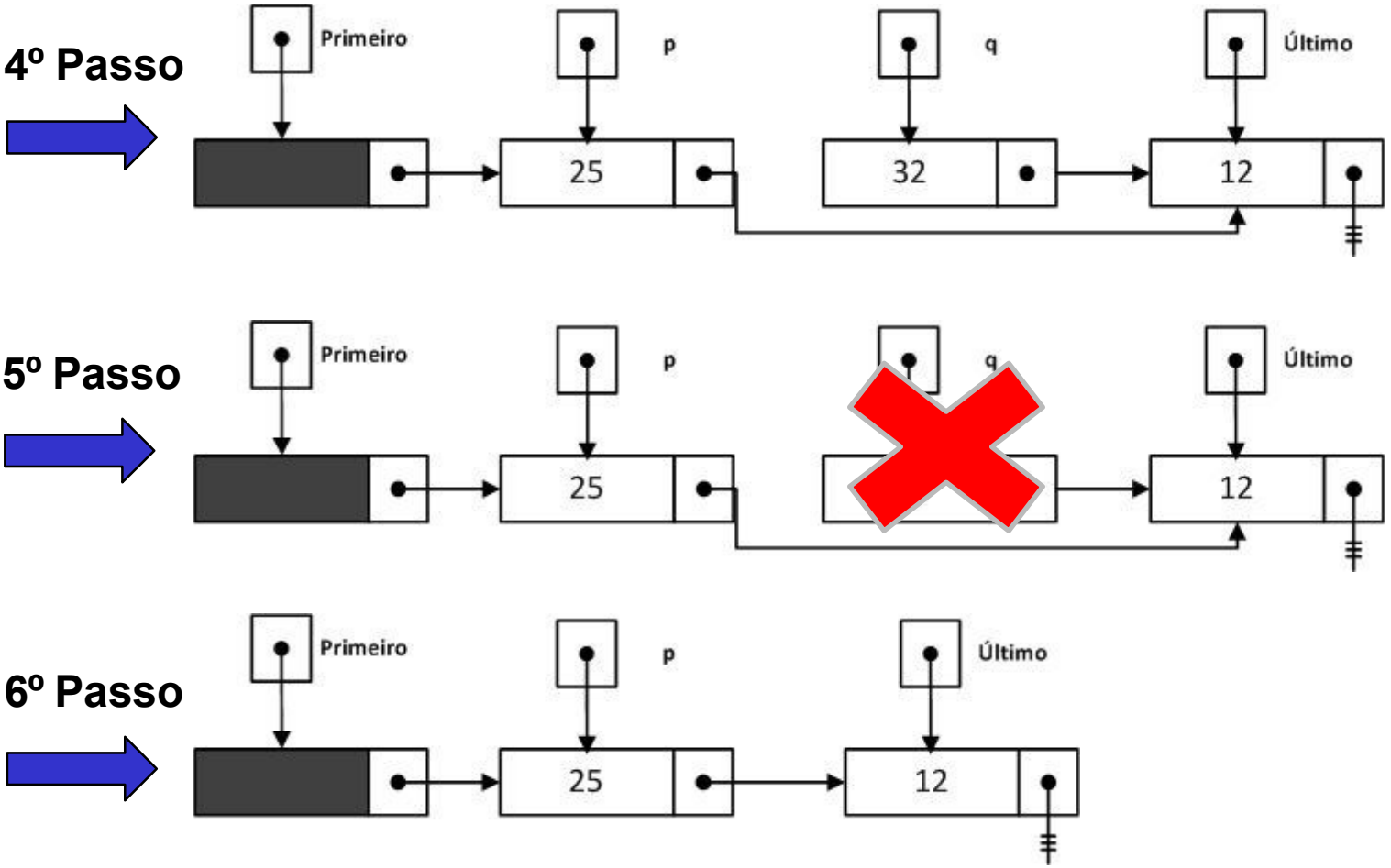
# Exercício (1)

- Seja nossa classe Lista, implemente um método que remove a segunda posição válida. Siga os passos da figura abaixo



# Exercício (1)

- Seja nossa classe Lista, implemente um método que remove a segunda posição válida. Siga os passos da figura abaixo



# Exercício (2)

- Crie uma classe ListaSimplesEncadeadaOrdenada, garantindo que os elementos sempre fiquem ordenados.

# Exercício (3)

- Na lista simples, crie um contador para a quantidade de elementos.

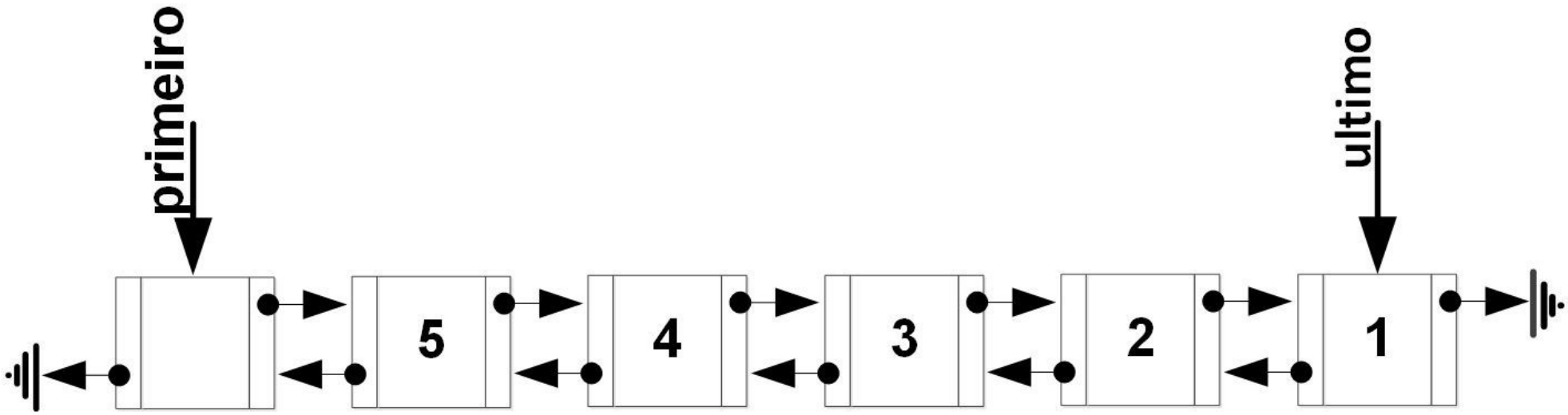
DICA: Nas inserções aumente o contador em uma unidade e nas remoções, decremente uma unidade.

# Exercício (4)

- Modifique o método *inserirInicio* de tal forma que o novo valor seja inserido no nó cabeça e, em seguida, criamos uma nova célula como nó cabeça

# Exercício (5)

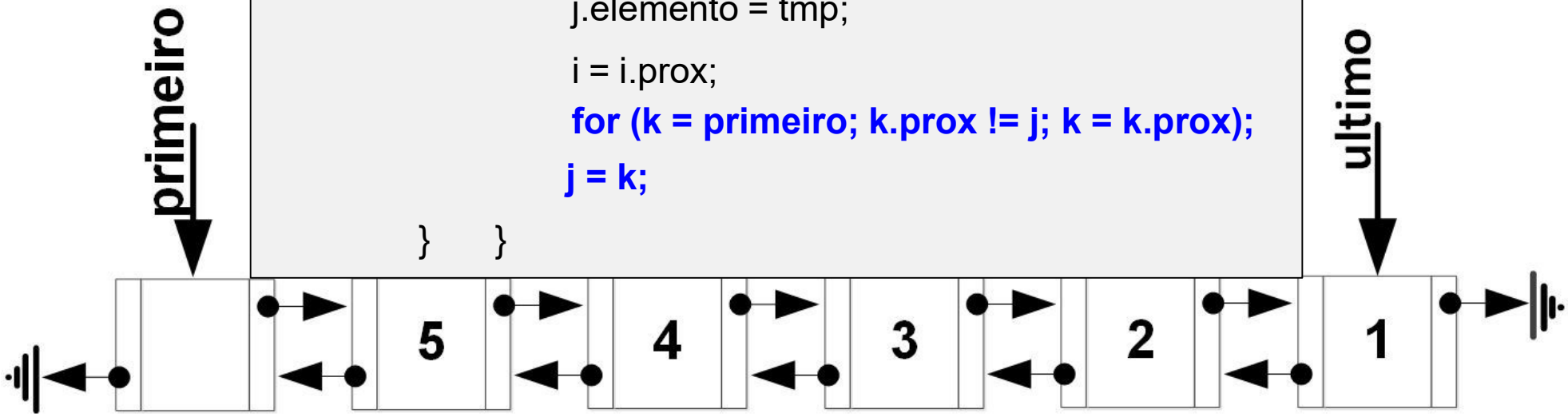
- Faça um método que inverta a ordem dos elementos da **lista simples**. No exemplo abaixo, após a inversão, os elementos ficarão na ordem crescente



# Exercício (6)

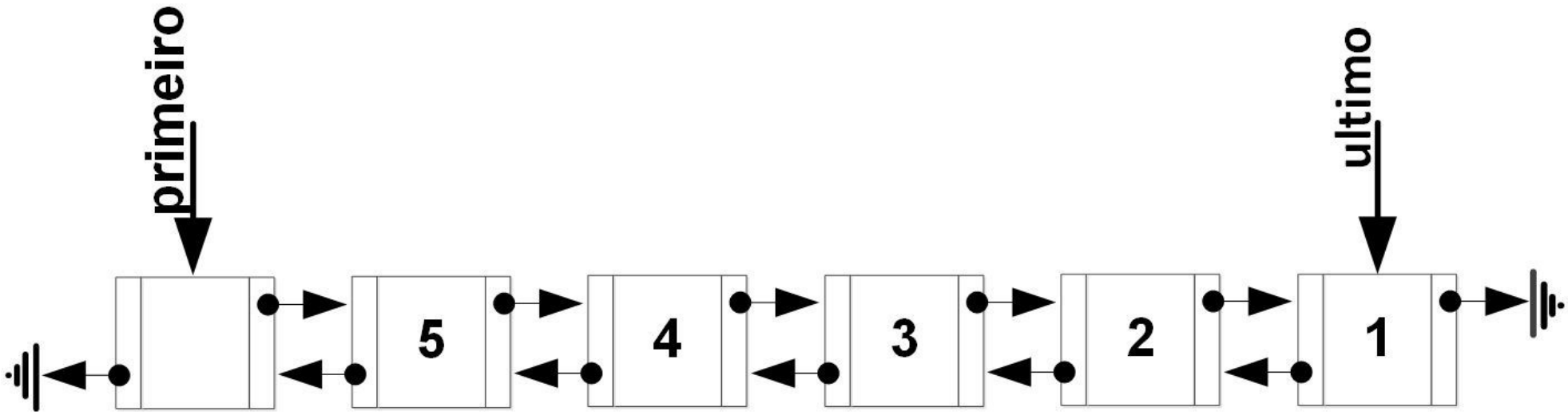
- Faça um método que inverta a ordem dos elementos da **lista simples**. No exemplo abaixo, após a inversão, os elementos ficarão na ordem crescente

```
void inverte(){
    Celula i = primeiro.prox;Celula j = ultimo;
    Celula k;
    while (i != j && j.prox != i){
        int tmp = i.elemento;
        i.elemento = j.elemento;
        j.elemento = tmp;
        i = i.prox;
        for (k = primeiro; k.prox != j; k = k.prox);
        j = k;
} }
```



# Exercício (7)

- Faça um método que inverta a ordem dos elementos da **lista dupla**. No exemplo abaixo, após a inversão, os elementos ficarão na ordem crescente

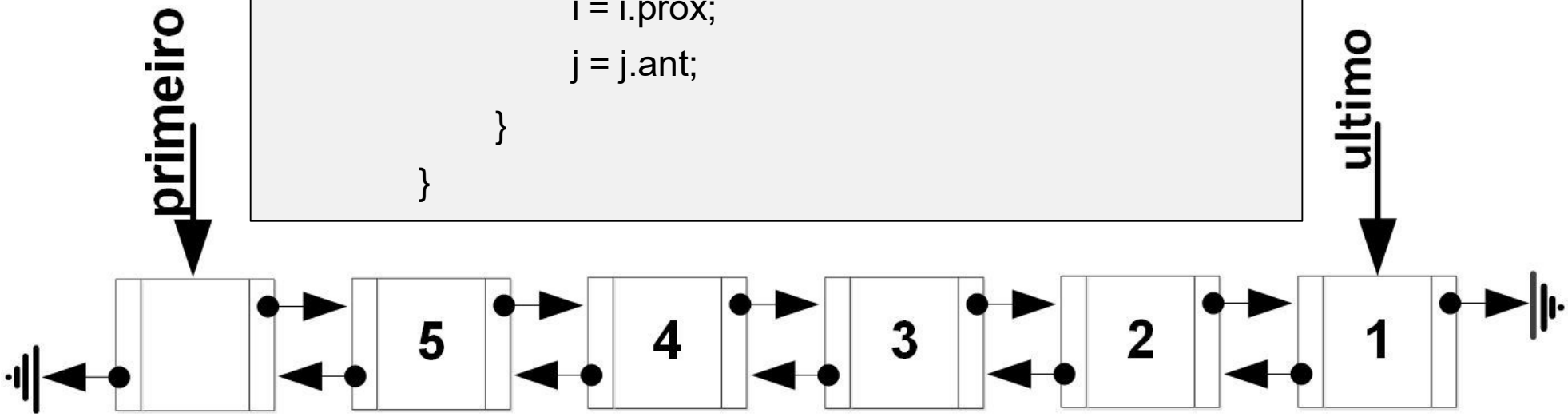


# Exercício (7)

- Faça um método que inverta a ordem dos elementos da **lista dupla**. No exemplo abaixo, após a inversão, os elementos ficarão na ordem crescente

```
void inverte(){
    Celula i = primeiro.prox;Celula j = ultimo;
    while (i != j && j.prox != i){
        int tmp = i.elemento;
        i.elemento = j.elemento;
        j.elemento = tmp;
        i = i.prox;
        j = j.ant;
    }
}
```

Veja que a condição **i != j** é para uma lista com número ímpar de elementos. Para par, **j.prox != i**.



# Exercícios (8)

- Faça um método que receba um nome de arquivos como parâmetro e leia *n* valores inteiros do arquivos e armazene somente os negativos na lista dupla.

# Exercício

- Implemente o algoritmo de Shellsort na **lista dupla**.

# Exercício

- Implemente o algoritmo de Shellsort na **lista simples**.

# Exercício

- Implemente o algoritmo de Quicksort na **lista dupla**.

# Exercício

- Implemente o algoritmo de Quicksort na **lista simples**.

# Próxima Aula

- Árvore Binária