

Estrutura de Dados e Algoritmos com Java

VETORES (ARRAYS)

<loiane.training />

02

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Introdução

VETORES (ARRAYS): INTRODUÇÃO

- Classe Vetor
 - Definição
 - Adicionar elemento final do vetor
 - Verificar quantidade de elementos no vetor
 - Imprimir elementos do vetor
 - Obter elemento de uma posição
 - Verificar se elemento existe no vetor
 - Adicionar elemento em qualquer posição
 - Adicionar mais capacidade ao vetor
 - Remover elemento do vetor
 - Generalizar o tipo dos elementos
 - Definindo o tipo do vetor dinamicamente
 - API Java ArrayList
 - Exercícios



Um vetor (ou array) é a estrutura de dados mais simples que existe.

Um vetor armazena uma sequência de valores onde todos são do mesmo tipo.

-Loiane Groner

Learning JavaScript Data Structures and Algorithms

ARMAZENAR TEMPERATURAS

```
double tempDia001 = 31.3;  
double tempDia002 = 32;  
double tempDia003 = 33.7;  
double tempDia004 = 34;  
double tempDia005 = 33.1;
```

ARMAZENAR TEMPERATURAS EM UM VETOR

```
double tempDia001 = 31.3;  
double tempDia002 = 32;  
double tempDia003 = 33.7;  
double tempDia004 = 34;  
double tempDia005 = 33.1;
```

```
double[] temperaturas = new double[365];  
temperaturas[0] = 31.3;  
temperaturas[1] = 32;  
temperaturas[2] = 33.7;  
temperaturas[3] = 34;  
temperaturas[4] = 33.1;
```

Início

31.3	32	33.7	34	33.1			
[0]	[1]	[2]	[3]	[4]	[5]	[...]	[364]

```
double[] temperaturas = new double[365];  
temperaturas[0] = 31.3;  
temperaturas[1] = 32;  
temperaturas[2] = 33.7;  
temperaturas[3] = 34;  
temperaturas[4] = 33.1;
```


AULA 19 DO CURSO DE JAVA BÁSICO

```
package com.loiane.cursojava.aula19;

public class Arrays {

    public static void main(String[] args) {

        double[] temperaturas = new double[365];
        temperaturas[0] = 31.3;
        temperaturas[1] = 32;
        temperaturas[2] = 33.7;
        temperaturas[3] = 34;
        temperaturas[4] = 33.1;

        System.out.println("O valor da temperatura do dia 3 é: " + temperaturas[2]);

        System.out.println("O tamanho do array: " + temperaturas.length);

        for (int i=0; i<temperaturas.length; i++){
            System.out.println("O valor da temperatura do dia " + (i+1) + " é: " +
temperaturas[i]);
        }

        for (double temp : temperaturas){
            System.out.println(temp);
        }
    }
}
```


AULA 19 DO CURSO DE JAVA BÁSICO

```
package com.loiane.cursojava.aula19;
```

```
public class Arrays {
```

```
    public static void main(String[] args) {
```

```
        double[] temperaturas = new double[365];
```

```
        temperaturas[0] = 31.3;
```

```
        temperaturas[1] = 32;
```

```
        temperaturas[2] = 33.7;
```

```
        temperaturas[3] = 34;
```

```
        temperaturas[4] = 33.1;
```

```
        System.out.println("O valor da temperatura do dia 3 é: " + temperaturas[2]);
```

```
        System.out.println("O tamanho do array: " + temperaturas.length);
```

```
        for (int i=0; i<temperaturas.length; i++){
```

```
            System.out.println("O valor da temperatura do dia " + (i+1) + " é: " +
```

```
temperaturas[i]);
```

```
        }
```

```
        for (double temp : temperaturas){
```

```
            System.out.println(temp);
```

```
        }
```

```
    }
```

```
}
```

Acessar posição X do array



`temperaturas[2];`

AULA 19 DO CURSO DE JAVA BÁSICO

```
package com.loiane.cursojava.aula19;
```

```
public class Arrays {
```

```
    public static void main(String[] args) {
```

```
        double[] temperaturas = new double[365];
```

```
        temperaturas[0] = 31.3;
```

```
        temperaturas[1] = 32;
```

```
        temperaturas[2] = 33.7;
```

```
        temperaturas[3] = 34;
```

```
        temperaturas[4] = 33.1;
```

Obter tamanho do array



```
        System.out.println("O valor da temperatura do dia 3 é: " + temperaturas[2]);
```

```
        System.out.println("O tamanho do array: " + temperaturas.length);
```

```
        for (int i=0; i<temperaturas.length; i++){
```

```
            System.out.println("O valor da temperatura do dia " + (i+1) + " é: " +
```

```
temperaturas[i]);
```

```
        }
```

```
        for (double temp : temperaturas){
```

```
            System.out.println(temp);
```

```
        }
```

```
    }
```

```
}
```

AULA 19 DO CURSO DE JAVA BÁSICO

```
package com.loiane.cursojava.aula19;
```

```
public class Arrays {
```

```
    public static void main(String[] args) {
```

```
        double[] temperaturas = new double[365];
```

```
        temperaturas[0] = 31.3;
```

```
        temperaturas[1] = 32;
```

```
        temperaturas[2] = 33.7;
```

```
        temperaturas[3] = 34;
```

```
        temperaturas[4] = 33.1;
```

Iterar todos os elementos com for

```
        System.out.println("O valor da temperatura do dia 3 é: " + temperaturas[2]);
```

```
        System.out.println("O tamanho do array: " + temperaturas.length);
```



```
        for (int i=0; i<temperaturas.length; i++){
            System.out.println("O valor da temperatura do dia " + (i+1) + " é: " +
temperaturas[i]);
        }
```

```
        for (double temp : temperaturas){
```

```
            System.out.println(temp);
```

```
        }
```

```
    }
```

```
}
```

AULA 19 DO CURSO DE JAVA BÁSICO

```
package com.loiane.cursojava.aula19;

public class Arrays {

    public static void main(String[] args) {

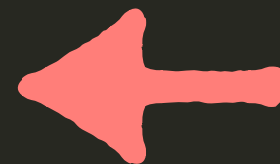
        double[] temperaturas = new double[365];
        temperaturas[0] = 31.3;
        temperaturas[1] = 32;
        temperaturas[2] = 33.7;
        temperaturas[3] = 34;
        temperaturas[4] = 33.1;

        System.out.println("O valor da temperatura do dia 3 é: " + temperaturas[2]);

        System.out.println("O tamanho do array: " + temperaturas.length);


        for (int i=0; i<temperaturas.length; i++){
            System.out.println("O valor da temperatura do dia " + (i+1) + " é: " +
temperaturas[i]);
        }

        for (double temp : temperaturas){
            System.out.println(temp);
        }
    }
}
```



*Iterar todos os elementos
com for melhorado*

CLASSE VETOR

- Definição 
- Adicionar elemento final do vetor
- Verificar quantidade de elementos no vetor
- Imprimir elementos do vetor
- Obter elemento de uma posição
- Verificar se elemento existe no vetor
- Adicionar elemento em qualquer posição
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

NOSSA CLASSE VETOR COMPLETA

```
package com.loiane.estruturados;

public class Vetor {

    private String[] elementos;

    public Vetor(int capacidade) {
        elementos = new String[capacidade];
    }

    public void adiciona(String elemento){ }

    public void adiciona(int posicao, String elemento){ }

    public void remove(int posicao){ }

    public String busca(int posicao){ }

    public int busca(String elemento){ }

    public int tamanho(){ }

    public String toString(){ }
}
```

NOSSA CLASSE VETOR COMPLETA

```
package com.loiane.estruturados;
```

```
public class Vetor {
```

```
    private String[] elementos;
```

```
    public Vetor(int capacidade) {  
        elementos = new String[capacidade];  
    }
```

```
    public void adiciona(String elemento){ }
```

```
    public void adiciona(int posicao, String elemento){ }
```

```
    public void remove(int posicao){ }
```

```
    public String busca(int posicao){ }
```

```
    public int busca(String elemento){ }
```

```
    public int tamanho(){ }
```

```
    public String toString(){ }
```

```
}
```

NOSSA CLASSE VETOR

```
package com.loiane.estruturados;  
  
public class Vetor {  
    private String[] elementos;  
  
    public Vetor(int capacidade) {  
        elementos = new String[capacidade];  
    }  
}
```


TESTE

```
package com.loiane.estruturados.aulas;

import com.loiane.estruturados.Vetor;

public class Aula02Vetor {

    public static void main(String[] args) {

        //vetor com capacidade para 100 elementos
        Vetor vetor = new Vetor(100);
    }
}
```

TESTE

.....

(x)= Variables ✕

●● Breakpoints

Name	Value
Ⓛ args	String[0] (id=15)
▼ Ⓛ vetor	Vetor (id=18)
▶ ■ elementos	String[100] (id=20)

[null, null, null, null, null, null, null, null, null, null, null, null

03

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Adicionar elemento final do vetor

ADICIONAR ELEMENTO FINAL DO VETOR – OPÇÃO 1

```
public void adiciona(String elemento){  
    for (int i=0; i<elementos.length; i++){  
        if (elementos[i] == null){  
            elementos[i] = elemento;  
            break;  
        }  
    }  
}
```


ADICIONAR ELEMENTO FINAL DO VETOR – OPÇÃO 1

```
public void adiciona(String elemento){  
    for (int i=0; i<elementos.length; i++){  
        if (elementos[i] == null){  
            elementos[i] = elemento;  
            break;  
        }  
    }  
}
```



ADICIONAR ELEMENTO FINAL DO VETOR 👍

```
package com.loiane.estruturados;

import java.util.Arrays;

public class Vetor {

    private String[] elementos;
    private int tamanho;

    public Vetor(int capacidade) {
        elementos = new String[capacidade];
        tamanho = 0;
    }

    public void adiciona(String elemento){
        elementos[tamanho] = elemento;
        tamanho++;
    }
}
```

ADICIONAR ELEMENTO FINAL DO VETOR 👍

```
package com.loiane.estruturados;
```

```
import java.util.Arrays;
```

```
public class Vetor {
```

```
    private String[] elementos;
```

```
    private int tamanho;
```

```
    public Vetor(int capacidade) {
```

```
        elementos = new String[capacidade];
```

```
        tamanho = 0;
```

```
    public void adiciona(String elemento){
```

```
        elementos[tamanho] = elemento;
```

```
        tamanho++;
```

```
    }
```

```
}
```

MELHORANDO O MÉTODO ADICIONA – OPÇÃO 1

```
public void adiciona(String elemento) throws Exception{
    if (tamanho < elementos.length){
        elementos[tamanho] = elemento;
        tamanho++;
    } else {
        throw new Exception("Vetor já está cheio, não é possível
adicionar mais elementos");
    }
}
```


MELHORANDO O MÉTODO ADICIONA – OPÇÃO 2

```
public boolean adiciona(String elemento) {  
    if (tamanho < elementos.length){  
        elementos[tamanho] = elemento;  
        tamanho++;  
        return true;  
    }  
    return false;  
}
```

TESTE

```
package com.loiane.estruturados.aulas;

import com.loiane.estruturados.Vetor;

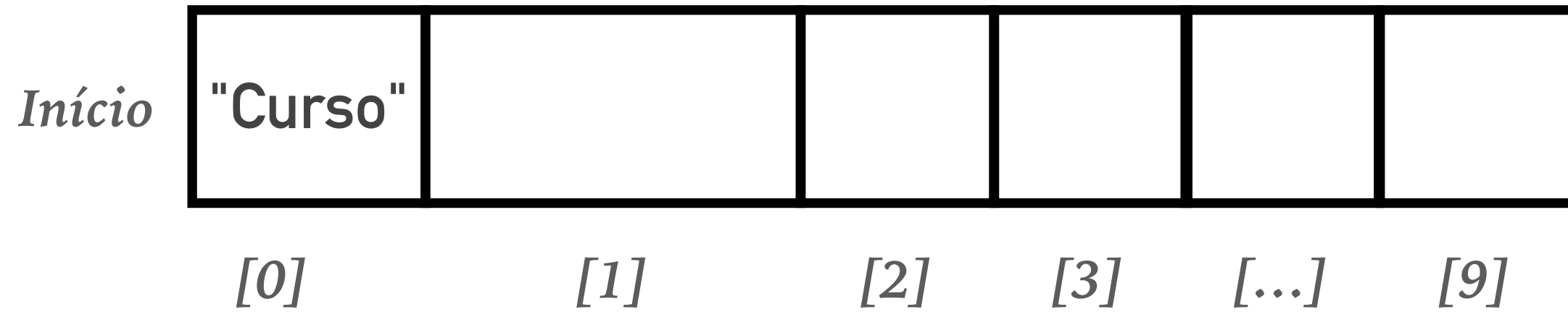
public class Aula03Vetor {

    public static void main(String[] args) {

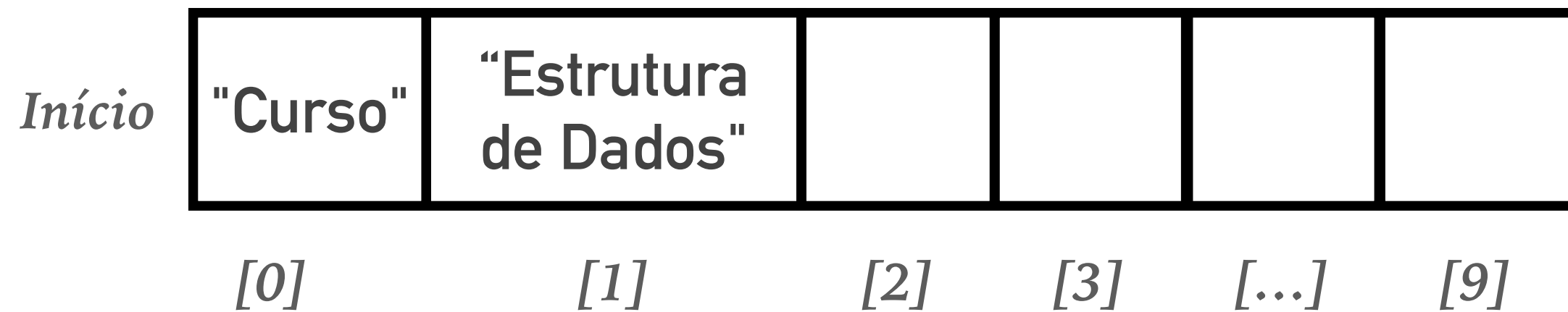
        //vetor com capacidade para 10 elementos
        Vetor vetor = new Vetor(10);

        vetor.adiciona("Curso");
        vetor.adiciona("Estrutura de Dados");
    }
}
```

tamanho = 0;



tamanho = 1;



tamanho = 2;

TESTE

.....



(x)= Variables ✕

●● Breakpoints

Name	Value
Ⓛ args	String[0] (id=15)
▼ Ⓛ vetor	Vetor (id=18)
▶ ■ elementos	String[10] (id=20)
■ tamanho	2

[Curso, Estrutura de Dados, null, null, null, null, null, null, null, null]

CLASSE VETOR

- Definição 
- Adicionar elemento final do vetor 
- Verificar quantidade de elementos no vetor
- Imprimir elementos do vetor
- Obter elemento de uma posição
- Verificar se elemento existe no vetor
- Adicionar elemento em qualquer posição
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

04

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Verificar tamanho e imprimir elementos

TESTE

```
package com.loiane.estruturados.aulas;

import com.loiane.estruturados.Vetor;

public class Aula03Vetor {

    public static void main(String[] args) {

        //vetor com capacidade para 10 elementos
        Vetor vetor = new Vetor(10);

        vetor.adiciona("Curso");
        vetor.adiciona("Estrutura de Dados");
    }
}
```


TESTE

.....

(x)= Variables ✕ Breakpoints	
Name	Value
args	String[0] (id=15)
▼ vetor	Vetor (id=18)
▶ elementos	String[10] (id=20)
tamanho	2
[Curso, Estrutura de Dados, null, null, null, null, null, null, null, null]	

VERIFICAR TAMANHO

```
public int tamanho(){  
    return this.tamanho;  
}
```

VERIFICAR TAMANHO

```
public int tamanho(){  
    return this.tamanho;  
}
```

*Pode ser método **get** também!*

*Mas não criamos o método **set** porque controlamos o tamanho apenas internamente*

IMPRIMIR OS ELEMENTOS DO VETOR

```
public String toString(){  
    return Arrays.toString(elementos);  
}
```

TESTE

```
//vetor com capacidade para 10 elementos  
Vetor vetor = new Vetor(10);
```

```
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");
```

```
System.out.print("Tamanho do vetor: ");  
System.out.println(vetor.tamanho());
```

```
System.out.println("Elementos do vetor:");  
System.out.println(vetor.toString());
```

TESTE

```
//vetor com capacidade para 10 elementos  
Vetor vetor = new Vetor(10);
```

```
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");
```

```
System.out.print("Tamanho do vetor: ");  
System.out.println(vetor.tamanho());
```

```
System.out.println("Elementos do vetor:");  
System.out.println(vetor.toString());
```

```
Tamanho do vetor: 2  
Elementos do vetor:  
[Curso, Estrutura de Dados, null, null, null, null, null, null, null, null]
```

IMPRIMIR OS ELEMENTOS DO VETOR

```
public String toString(){  
    return Arrays.toString(elementos);  
}
```



IMPRIMIR OS ELEMENTOS DO VETOR

```
public String toString(){
    StringBuilder s = new StringBuilder();
    s.append("[");

    for (int i=0; i<this.tamanho-1; i++){
        s.append(elementos[i]);
        s.append(", ");
    }

    if (this.tamanho>0){
        s.append(elementos[this.tamanho-1]);
    }
    s.append("]");

    return s.toString();
}
```

TESTE

```
//vetor com capacidade para 10 elementos  
Vetor vetor = new Vetor(10);
```

```
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");
```

```
System.out.print("Tamanho do vetor: ");  
System.out.println(vetor.tamanho());
```

```
System.out.println("Elementos do vetor:");  
System.out.println(vetor.toString());
```

```
Tamanho do vetor: 2  
Elementos do vetor:  
[Curso, Estrutura de Dados]
```

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição
- Verificar se elemento existe no vetor
- Adicionar elemento em qualquer posição
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios



Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Obter elemento de uma posição

OBTER ELEMENTO DE UMA POSIÇÃO

```
public String busca(int posicao){  
    if (!(posicao >= 0 && posicao < tamanho)){  
        throw new IllegalArgumentException("Posição inválida")  
    }  
    return elementos[posicao];  
}
```

TESTE

```
Vetor vetor = new Vetor(10);  
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");  
  
System.out.print("Elemento da posição 1: ");  
System.out.println(vetor.busca(1));  
  
System.out.print("Elemento da posição 3: ");  
System.out.println(vetor.busca(3));
```


TESTE

```
Vetor vetor = new Vetor(10);  
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");  
  
System.out.print("Elemento da posição 1: ");  
System.out.println(vetor.busca(1));  
  
System.out.print("Elemento da posição 3: ");  
System.out.println(vetor.busca(3));
```

Elemento da posição 1: Estrutura de Dados

Elemento da posição 3:

Exception in thread "main" java.lang.IllegalArgumentException: Posicao inválida
at com.loiane.estruturadados.Vetor.busca(Vetor.java:43)
at com.loiane.estruturadados.aulas.Aula05Vetor.main(Aula05Vetor.java:19)

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor
- Adicionar elemento em qualquer posição
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios



Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Verificar se elemento existe no vetor

VERIFICAR SE ELEMENTO EXISTE == BUSCA SEQUENCIAL

```
public int busca(String elemento){  
    for (int i=0; i<tamanho ;i++){  
        if (elementos[i].equals(elemento)){  
            return i;  
        }  
    }  
    return -1;  
}
```

TESTE

```
Vetor vetor = new Vetor(10);  
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");  
  
System.out.print("Busca elemento 'Estrutura de Dados': ");  
System.out.println(vetor.busca("Estrutura de Dados"));  
  
System.out.print("Busca elemento 'loiane.training': ");  
System.out.println(vetor.busca("loiane.training"));
```

TESTE

```
Vetor vetor = new Vetor(10);  
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");  
  
System.out.print("Busca elemento 'Estrutura de Dados': ");  
System.out.println(vetor.busca("Estrutura de Dados"));  
  
System.out.print("Busca elemento 'loiane.training': ");  
System.out.println(vetor.busca("loiane.training"));
```

```
Busca elemento 'Estrutura de Dados': 1  
Busca elemento 'loiane.training': -1
```

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

07

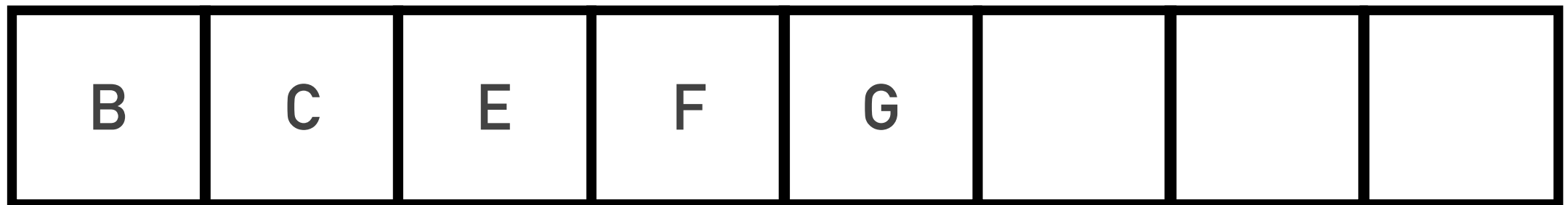
Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Adicionar elemento em qualquer posição

Início



[0]

[1]

[2]

[3]

[4]

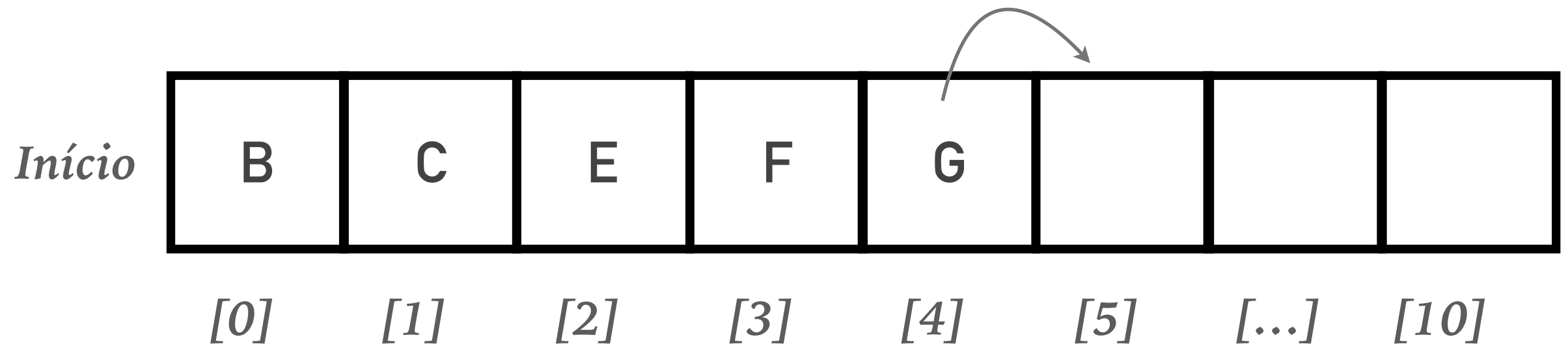
[5]

[...]

[10]

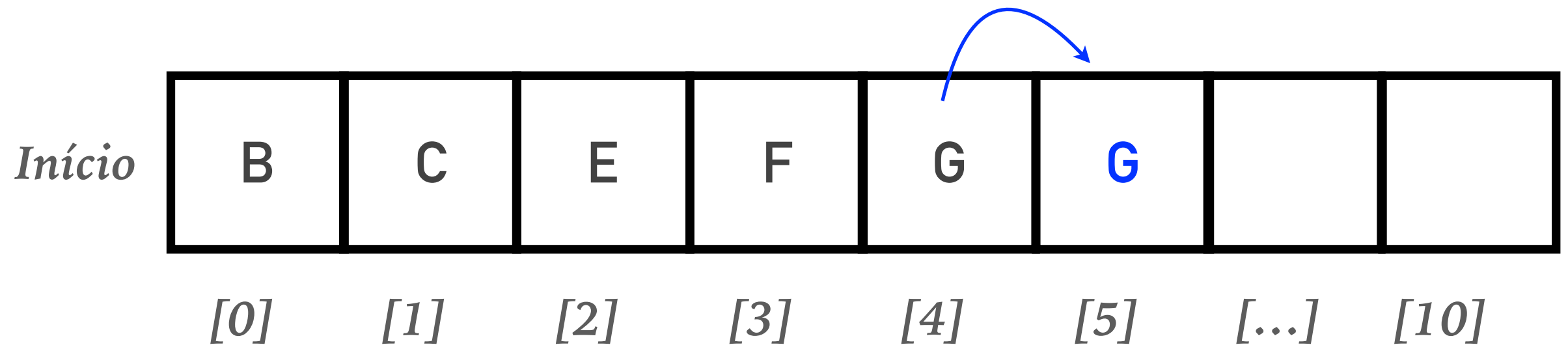
tamanho = 5

Inserir "A" na posição 0 do vetor



tamanho: 5

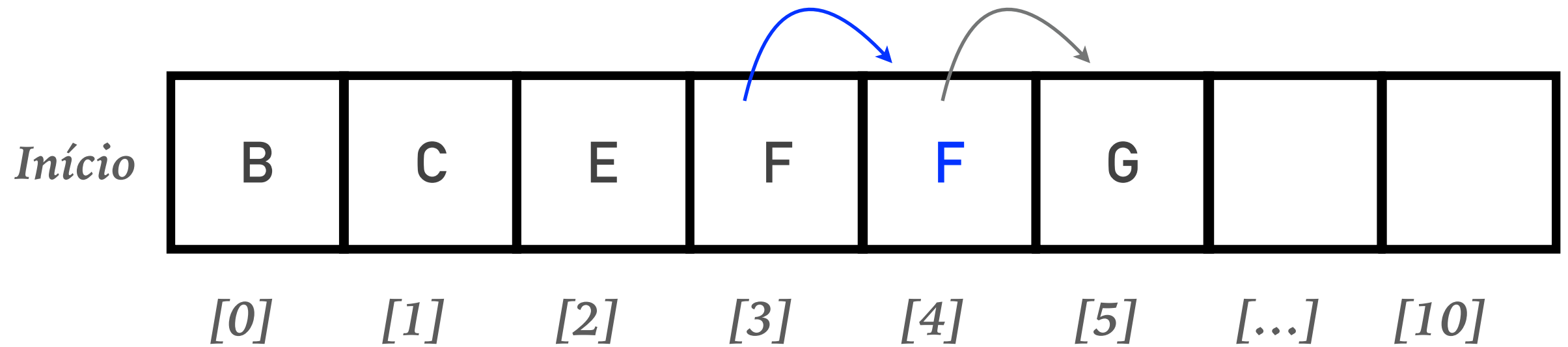
posição a ser inserido: 0



tamanho: 5

posição a ser inserido: 0

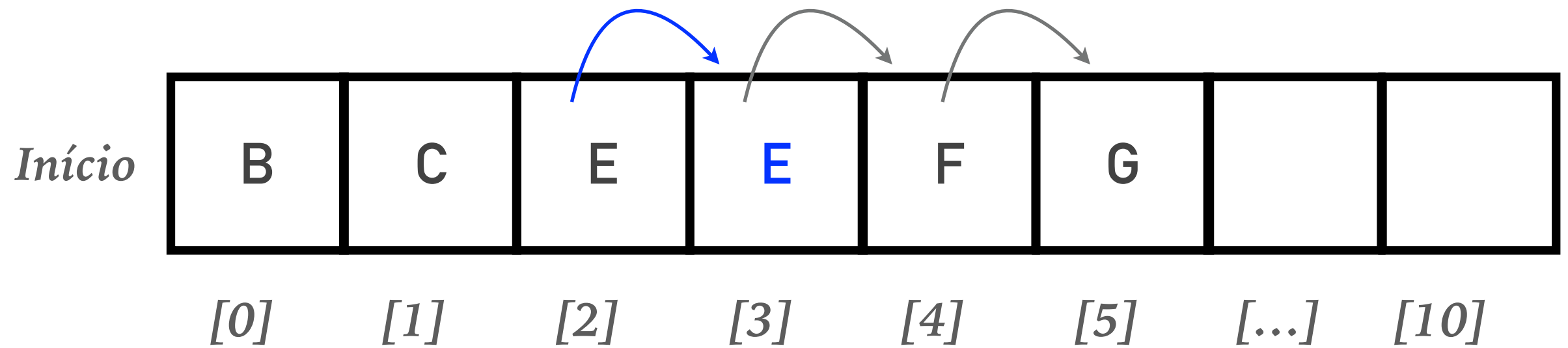
$\text{vetor}[5] = \text{vetor}[4]$



tamanho: 5

posição a ser inserido: 0

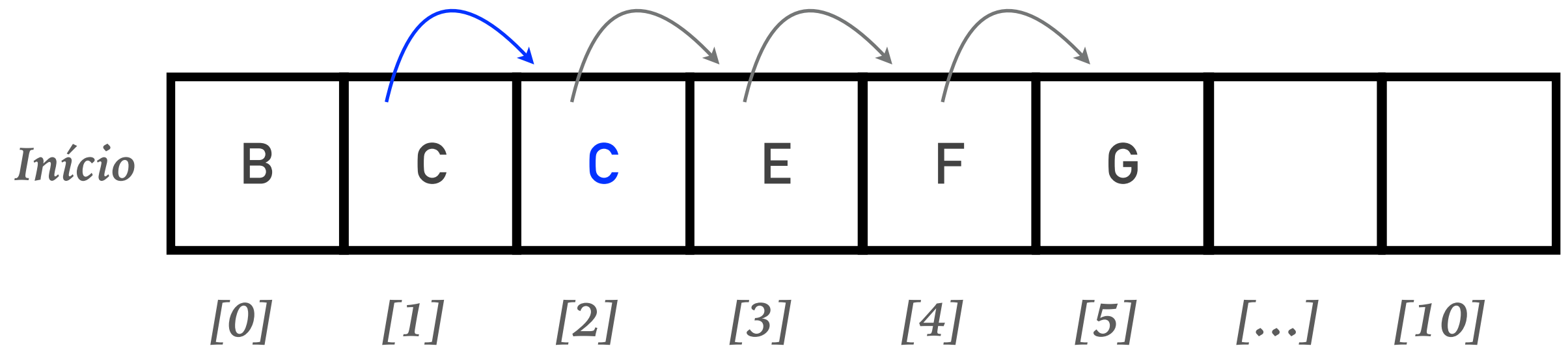
$\text{vetor}[4] = \text{vetor}[3]$



tamanho: 5

posição a ser inserido: 0

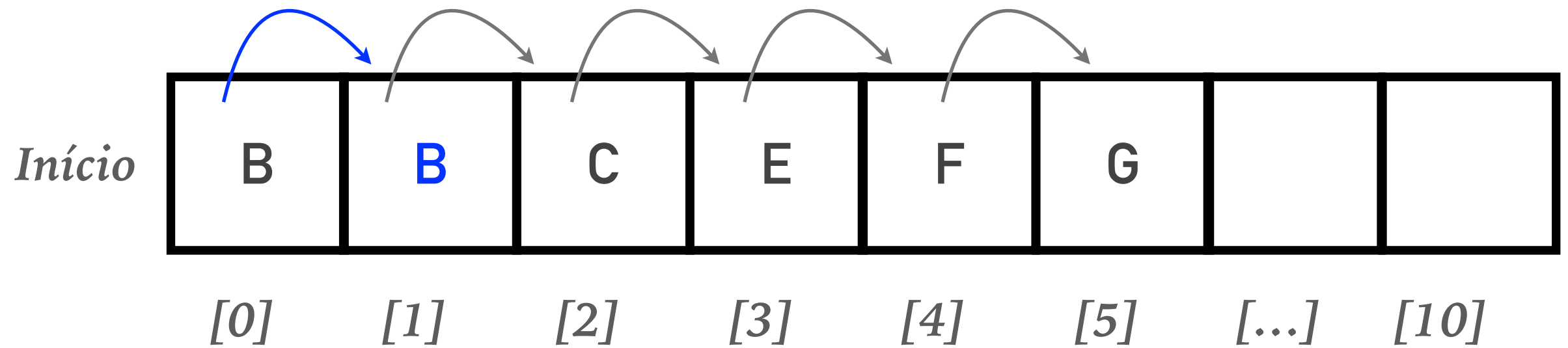
$\text{vetor}[3] = \text{vetor}[2]$



tamanho: 5

posição a ser inserido: 0

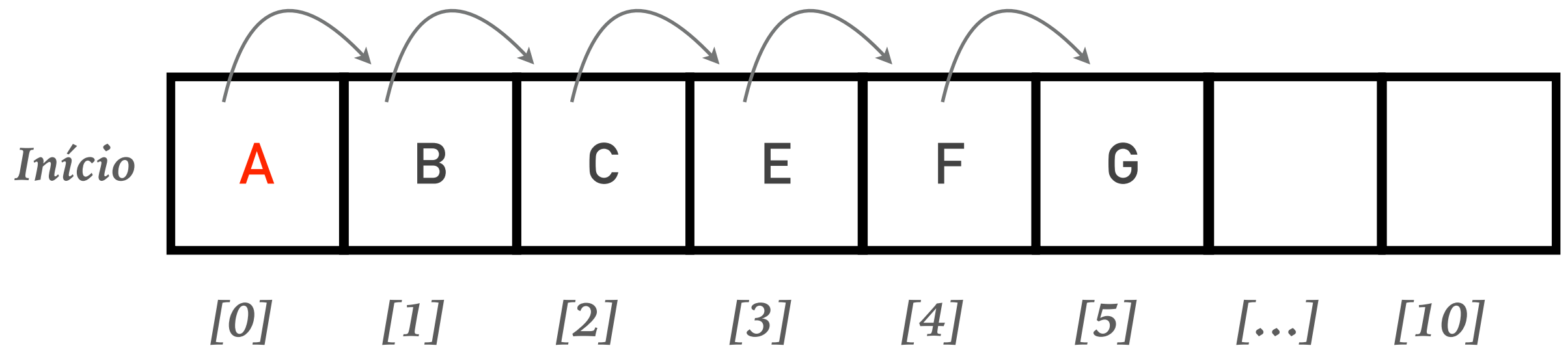
$\text{vetor}[2] = \text{vetor}[1]$



tamanho: 5

posição a ser inserido: 0

$vetor[1] = vetor[0]$



tamanho: 6

posição a ser inserido: 0

vetor[0] = "A"

ADICIONAR ELEMENTO EM QUALQUER POSIÇÃO

```
public void adiciona(int posicao, String elemento){  
    if (!(posicao >= 0 && posicao <= tamanho)){  
        throw new IllegalArgumentException("Posicao inválida");  
    }  
    for (int i=tamanho-1; i>=posicao; i--){  
        elementos[i+1] = elementos[i];  
    }  
  
    elementos[posicao] = elemento;  
    tamanho++;  
}
```

TESTE

```
Vetor vetor = new Vetor(10);  
  
vetor.adiciona("Curso");  
vetor.adiciona("Estrutura de Dados");  
vetor.adiciona(0, "http://loiane.training");  
vetor.adiciona(3, "http://loiane.com");  
  
System.out.println(vetor);
```

[http://loiane.training, Curso, Estrutura de Dados, http://loiane.com]

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios



Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Adicionar capacidade ao vetor

ADICIONAR CAPACIDADE AO VETOR DINAMICAMENTE

```
private void aumentaCapacidade(){
    if (tamanho == elementos.length){
        String[] elementosNovos = new String[elementos.length * 2];
        for (int i=0; i<elementos.length; i++){
            elementosNovos[i] = elementos[i];
        }
        elementos = elementosNovos;
    }
}
```

ADICIONAR ELEMENTO NO FINAL DO VETOR

```
public boolean adiciona(String elemento) {  
    aumentaCapacidade();  
    if (tamanho < elementos.length) {  
        elementos[tamanho] = elemento;  
        tamanho++;  
        return true;  
    }  
    return false;  
}
```

ADICIONAR ELEMENTO EM QUALQUER POSIÇÃO

```
public void adiciona(int posicao, String elemento){  
    aumentaCapacidade();  
    if (!(posicao >= 0 && posicao <= tamanho)){  
        throw new IllegalArgumentException("Posicao inválida");  
    }  
    for (int i=tamanho-1; i>=posicao; i--){  
        elementos[i+1] = elementos[i];  
    }  
  
    elementos[posicao] = elemento;  
    tamanho++;  
}
```


TESTE

```
Vetor vetor = new Vetor(5);
```

```
vetor.adiciona("Curso");
```

```
vetor.adiciona("Estrutura de Dados");
```

```
vetor.adiciona(0, "http://loiane.training");
```

```
vetor.adiciona(3, "http://loiane.com");
```

```
vetor.adiciona("Java");
```

```
vetor.adiciona("Vetor");
```

```
vetor.adiciona("Array");
```

```
System.out.println(vetor);
```

```
System.out.println(vetor.tamanho());
```

[http://loiane.training, Curso, Estrutura de Dados, http://loiane.com, Java, Vetor, Array]

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor ✓
- Remover elemento do vetor
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

09

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Remover elemento do vetor

Início

B	G	D	E	F			
---	---	---	---	---	--	--	--

[0]

[1]

[2]

[3]

[4]

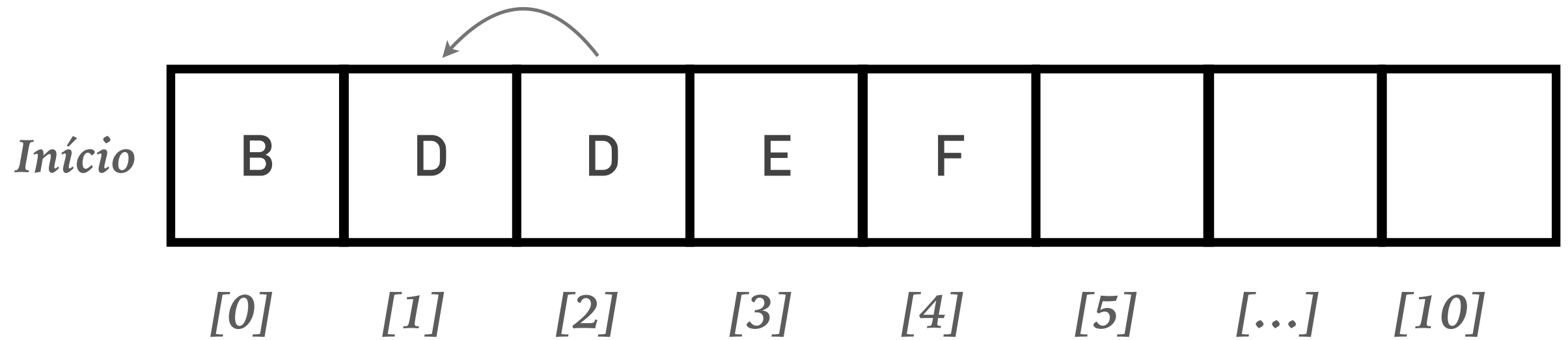
[5]

[...]

[10]

tamanho = 5

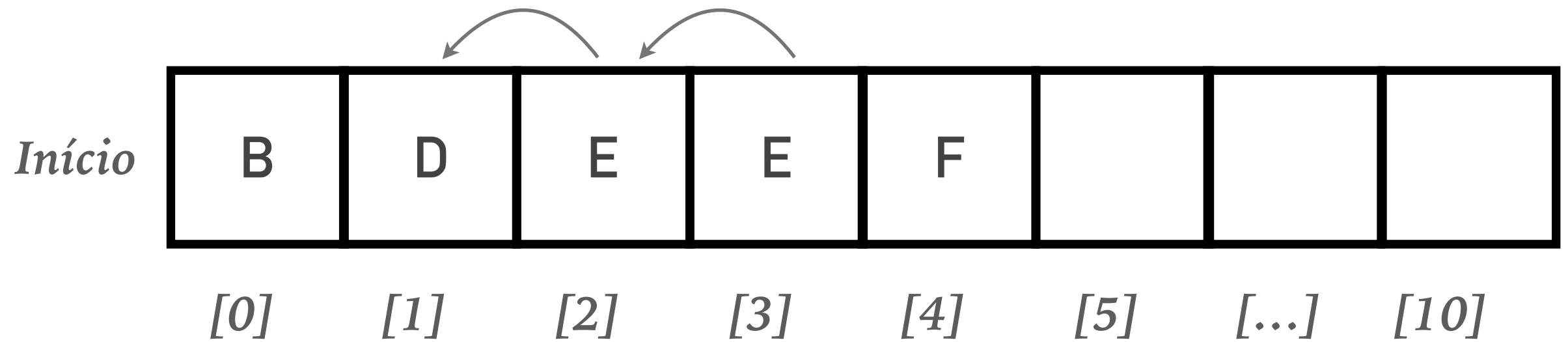
Remover “G” posição 1



tamanho: 5

posição a ser removida: 1

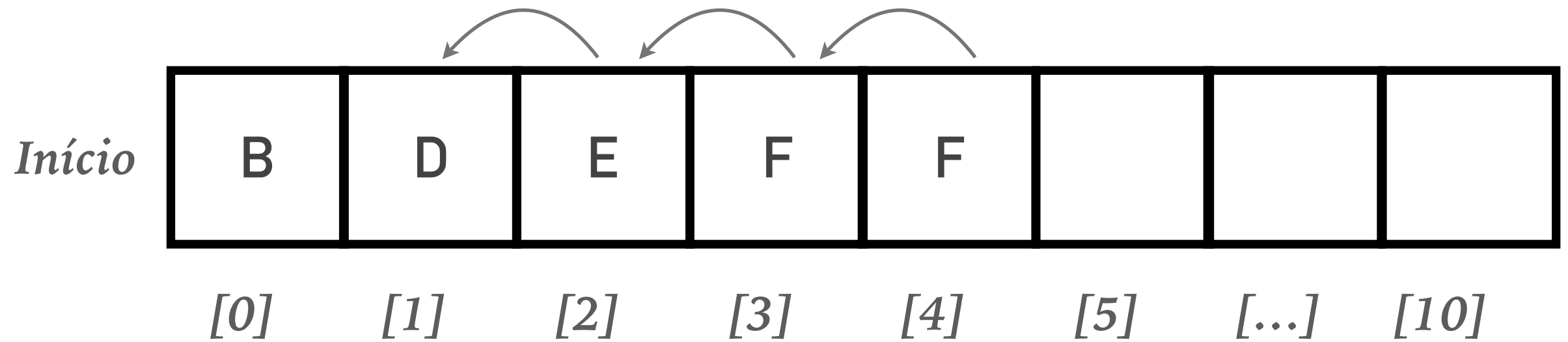
$\text{vetor}[1] = \text{vetor}[2]$



tamanho: 5

posição a ser removida: 1

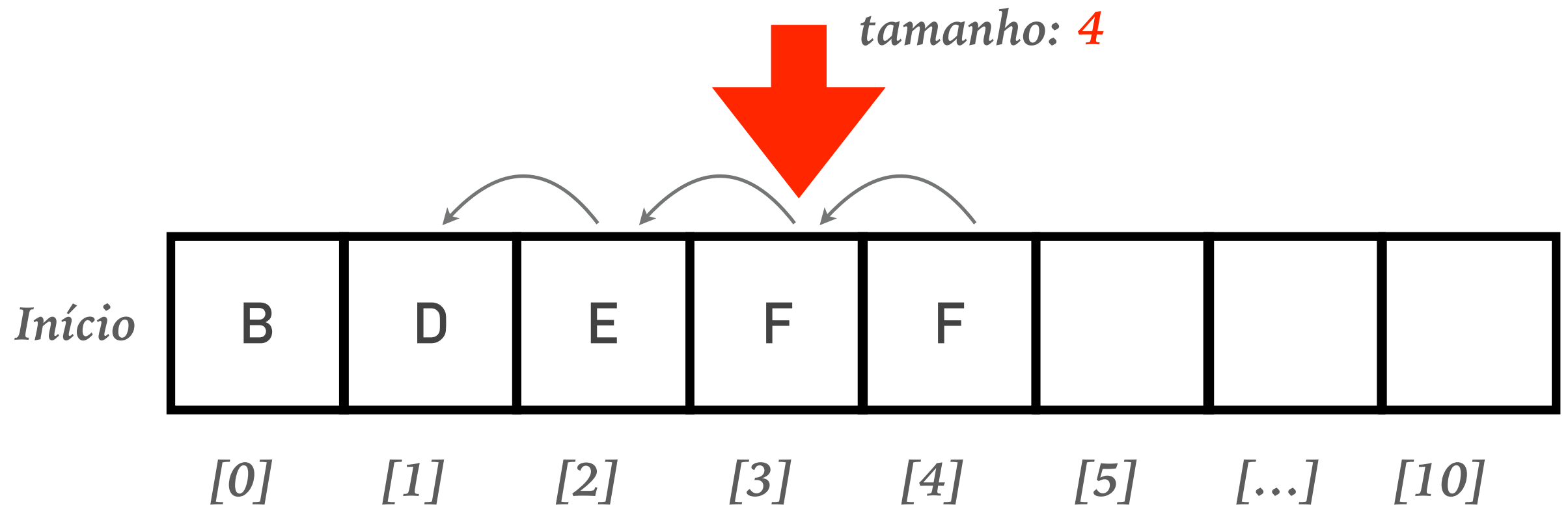
$\text{vetor}[2] = \text{vetor}[3]$



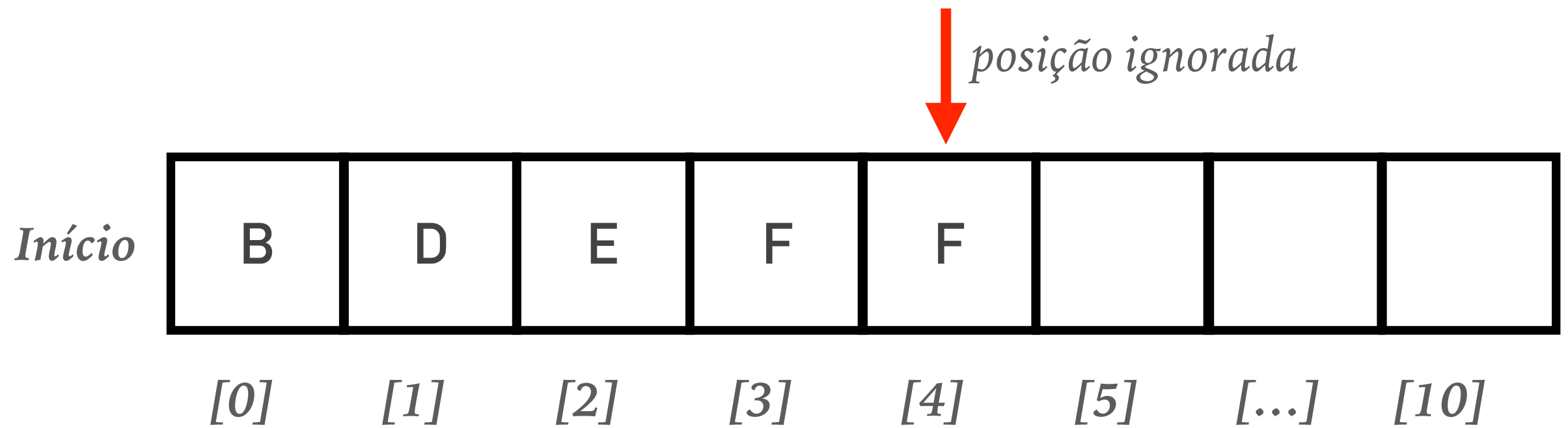
tamanho: 5

posição a ser removida: 1

$\text{vetor}[3] = \text{vetor}[4]$



tamanho: 4



tamanho: 4

posição a ser removida: 1

REMOVER ELEMENTO DE UMA POSIÇÃO

```
public void remove(int posicao){
    if (!(posicao >= 0 && posicao < tamanho)){
        throw new IllegalArgumentException("Posicao inválida");
    }
    for (int i=posicao; i<tamanho-1; i++){
        elementos[i] = elementos[i+1];
    }
    tamanho--;
}
```

REMOVER ELEMENTO DO VETOR

```
public int busca(String elemento)
public void remove(int posicao)
```

TESTE

```
Vetor vetor = new Vetor(5);
```

```
vetor.adiciona("Curso");
```

```
vetor.adiciona("Estrutura de Dados");
```

```
vetor.adiciona(0, "http://loiane.training");
```

```
vetor.adiciona(3, "http://loiane.com");
```

```
vetor.adiciona("Java");
```

```
vetor.adiciona("Vetor");
```

```
vetor.adiciona("Array");
```

```
System.out.println(vetor.tamanho());
```

```
vetor.remove(5);
```

```
vetor.remove(4);
```

```
vetor.remove(0);
```

```
System.out.println(vetor);
```

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor ✓
- Remover elemento do vetor ✓
- Generalizar o tipo dos elementos
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

10

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Generalizando o tipo do vetor

VETOR DE OBJECT

```
public class VetorObjetos {  
  
    private Object[] elementos;  
    private int tamanho;  
  
    public VetorObjetos(int capacidade) {  
        elementos = new Object[capacidade];  
        tamanho = 0;  
    }  
  
    //outros métodos  
}
```

CLASSE CONTATO

```
public class Contato {  
  
    private String nome;  
    private String telefone;  
    private String email;  
  
    // construtores  
    // métodos get e set  
    // toString e equals  
}
```


TESTE

.....

```
VetorObjetos vetor = new VetorObjetos(5);
```

```
Contato c1 = new Contato("Contato 1", "1234-5678", "contato1@email.com");  
Contato c2 = new Contato("Contato 2", "2345-6789", "contato2@email.com");  
Contato c3 = new Contato("Contato 3", "3456-7890", "contato3@email.com");
```

```
vetor.adiciona(c1);  
vetor.adiciona(1, c2);  
vetor.adiciona(c3);
```

```
System.out.println(vetor);
```

```
[Contato [nome=Contato 1, telefone=1234-5678, email=contato1@email.com],  
Contato [nome=Contato 2, telefone=2345-6789, email=contato2@email.com],  
Contato [nome=Contato 3, telefone=3456-7890, email=contato3@email.com]]
```

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor ✓
- Remover elemento do vetor ✓
- Generalizar o tipo dos elementos ✓
- Definindo o tipo do vetor dinamicamente
- API Java ArrayList
- Exercícios

11

Estrutura de Dados e Algoritmos com Java

<loiane.training />

VETORES (ARRAYS)

Definindo o tipo do Vetor dinamicamente

VETOR COM GENERICS

```
/**
 * @param <T> tipo do vetor/lista
 */
public class Lista<T> {

    private T[] elementos;
    private int tamanho;

    public Lista(int capacidade) {
        elementos = (T[]) new Object[capacidade];
        tamanho = 0;
    }

    //segunda opção de instancia vetor com Generics
    public Lista(int capacidade, Class<T> tipoClasse) {
        elementos = (T[]) Array.newInstance(tipoClasse, capacidade);
        tamanho = 0;
    }

    public boolean adiciona(T elemento) {}
    public int busca(T elemento){}
    //outros métodos
}
```

TESTE

.....

```
Lista<Contato> lista = new Lista<Contato>(5);
```

```
Contato c1 = new Contato("Contato 1", "1234-5678", "contato1@email.com");  
Contato c2 = new Contato("Contato 2", "2345-6789", "contato2@email.com");  
Contato c3 = new Contato("Contato 3", "3456-7890", "contato3@email.com");
```

```
lista.adiciona(c1);  
lista.adiciona(1, c2);  
lista.adiciona(c3);
```

```
//lista.adiciona("Uma String"); //erro de compilação
```

```
System.out.println(lista);
```

```
[Contato [nome=Contato 1, telefone=1234-5678, email=contato1@email.com],  
Contato [nome=Contato 2, telefone=2345-6789, email=contato2@email.com],  
Contato [nome=Contato 3, telefone=3456-7890, email=contato3@email.com]]
```


CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor ✓
- Remover elemento do vetor ✓
- Generalizar o tipo dos elementos ✓
- Definindo o tipo do vetor dinamicamente ✓
- API Java ArrayList
- Exercícios

12

Estrutura de Dados e Algoritmos com Java

<loiane.training />

API JAVA: ARRAYLIST

Usando a classe ArrayList do Java

TESTE 1

.....

```
ArrayList<Contato> lista = new ArrayList<Contato>(5);
```

```
Contato c1 = new Contato("Contato 1", "1234-5678", "contato1@email.com");  
Contato c2 = new Contato("Contato 2", "2345-6789", "contato2@email.com");  
Contato c3 = new Contato("Contato 3", "3456-7890", "contato3@email.com");
```

```
lista.add(c1);  
lista.add(1, c2);  
lista.add(c3);
```

```
//lista.add("Uma String"); //erro de compilação
```

```
System.out.println(lista);  
System.out.println(lista.size());
```

```
[Contato [nome=Contato 1, telefone=1234-5678, email=contato1@email.com],  
Contato [nome=Contato 2, telefone=2345-6789, email=contato2@email.com],  
Contato [nome=Contato 3, telefone=3456-7890, email=contato3@email.com]]
```


TESTE 2

.....

```
boolean contatoExiste = lista.contains(c1);
System.out.println("Contato 1 existe na lista");

int index = lista.indexOf(c1);
System.out.println("Contato 1 existe na posição " + index);

lista.remove(2);

System.out.println(lista);
System.out.println(lista.size());
```

Contato 1 existe na lista

Contato 1 existe na posição 0

[Contato [nome=Contato 1, telefone=1234-5678, email=contato1@email.com],

Contato [nome=Contato 2, telefone=2345-6789, email=contato2@email.com]]

2

CLASSE VETOR

- Definição ✓
- Adicionar elemento final do vetor ✓
- Verificar quantidade de elementos no vetor ✓
- Imprimir elementos do vetor ✓
- Obter elemento de uma posição ✓
- Verificar se elemento existe no vetor ✓
- Adicionar elemento em qualquer posição ✓
- Adicionar mais capacidade ao vetor ✓
- Remover elemento do vetor ✓
- Generalizar o tipo dos elementos ✓
- Definindo o tipo do vetor dinamicamente ✓
- API Java ArrayList ✓
- Exercícios



EXERCÍCIOS

<http://goo.gl/0FheJF>



Estrutura de Dados

Estrutura de Dados e Algoritmos com Java

GRÁTIS

INICIAR CURSO

HOME

O QUE VAMOS APRENDER NESSE CURSO?

- Vetores (Arrays)
- Pilhas (Stacks)
- Filas (Queues)
- Listas Encadeadas (Linked Lists)
- Listas Duplamente Encadeadas (Doubly-Linked Lists)
- Conjuntos (Sets)
- Tabelas de Hashing (HashTables)
- Árvores (Trees)
- Grafos (Graphs)
- Algoritmos de Ordenação:
 - Bolha (Bubble Sort)

Download código fonte e certificado
Cadastro em:

<http://loiane.training>

Obrigada



<http://loiane.com>



facebook.com/loianegroner



[@loiane](https://twitter.com/loiane)



<https://github.com/loiane>



youtube.com/user/Loianeg