PROGRAMAÇÃO ORIENTADA A OBJETOS

AULA 8 – VETORES, MATRIZES E ARRAYLIST

WALTER TRAVASSOS SARINHO

@PROF.WALTERTRAVASSOS

WALTER.TRAVASSOS@SECTRAS.EDU.BR

Dúvidas sobre a aula passada

Aula de Hoje

- Vetores (for each).
- Matrizes.
- Classe ArrayList.

ENQUANTO ISSO, NO CLARIM DIÁRIO...

A declaração de várias variáveis, uma a uma, é suficiente para codificar alguns tipos de programas.

Porém, este recurso não é suficiente para resolver **TODOS** os problemas computacionais...

ou é?

Mapa de Estudos

AULA 01
Paradigmas de Programação e história do Java.
IDEs e Hello Universe

AULA 03
Visibilidade de variáveis, classe String e casting.

AULA 02
Manipulação de datas.
Sintaxe da Linguagem Java e variáveis primitivas e de referência.

AULA 04
Classes Wrapper.
Operadores e métodos de conversão.

AULA 05
Formatar a saída de dados numérica.
Operador condicional if.
Métodos para comparar Strings

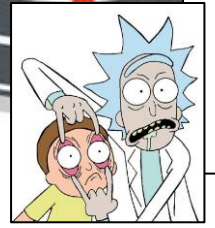
Avaliação 1

Java

Mapa de Estudos

AULA 06
Manipulação de Strings. Operador condicional switch.

AULA 07
Operadores de atribuição compostos. Estruturas de repetição: while, do while e for.



AULA 08
Vetores (for each) e matrizes. Classe ArrayList

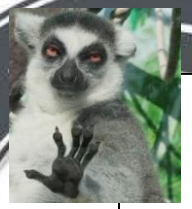
AULA 09
O paradigma Orientado a Objetos

AULA 10
Construindo métodos e Métodos construtores

Avaliação 2

Java

Mapa de Estudos



AULA 12
Classes Abstratas e Interfaces

AULA 11
Pilares da POO: Abstração, Encapsulamento, Herança e Polimorfismo

Avaliação 3

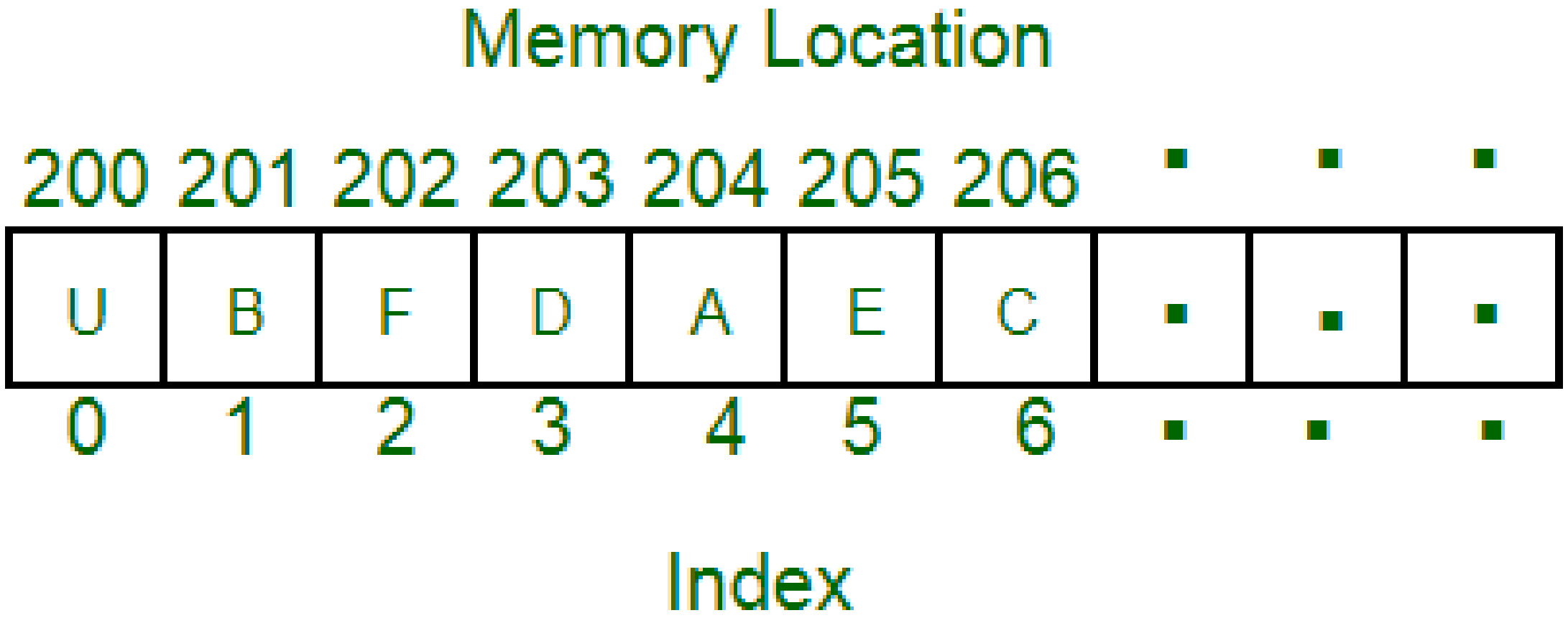
Java

Por exemplo...

- Faça um programa que leia o nome de 500 pessoas e, em seguida, escreva o nome de cada uma.

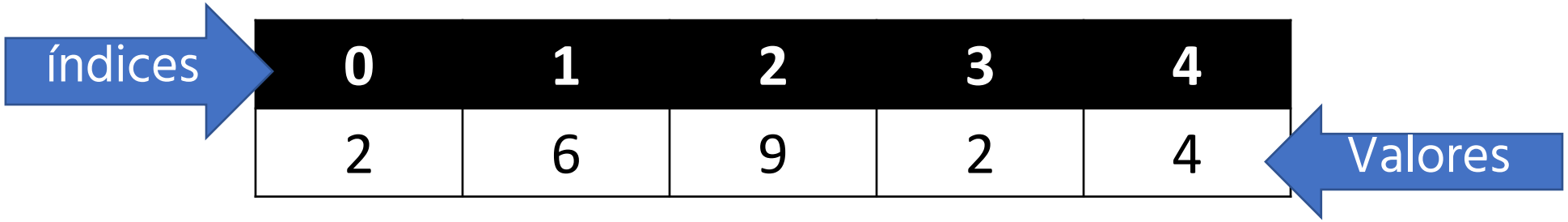
- Você iria declarar e utilizar 500 variáveis? Trabalhoso, não?!
- Só que, para resolver problemas similares a este, utilizamos vetores e matrizes (arrays).

Variáveis Compostas Homogêneas
Vetores ou Arranjos (Arrays)



Vetores

- Vários valores podem ser armazenados em uma única variável, chamada vetor.
- Vetores possuem índices por onde acessamos os valores armazenados.
- Vetores funcionam como uma tabela. Acessamos cada item (valor) por meio de suas colunas (índices).



Bacana...
Mas como eu crio um Vetor em Java?

Declaração do Vetor

- Exemplo de declaração de um vetor:

```
int idades[] = new int[10];
```

ou

```
int[] idades = new int[10];
```

- Os colchetes após o identificador da variável indicam a declaração de um vetor. É possível associar o colchete ao tipo da variável também.
- É **preciso definir o tamanho do vetor** ([10]) e **alocar memória** para o armazenamento dos elementos (new int).

Para utilizar um vetor...

1. Declare um array: **int[] v1;**

2. Inicialize o array e defina o tamanho: **v1 = new int[3];**

   int[] v1 = new int[3];

3. Por fim, armazene os valores dentro do array:

   v1[0] = 2;

   v1[1] = 3;

   v1[2] = 7;

Como atribuir valores a um vetor?

3 minutos

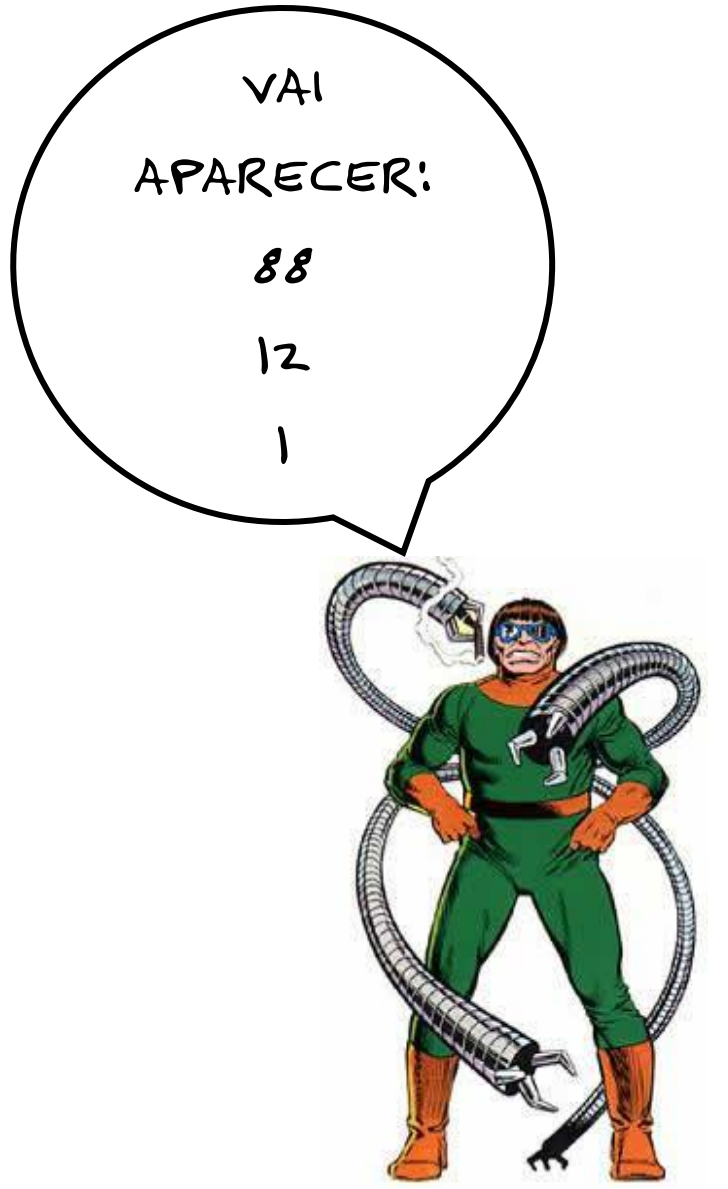
Devemos fornecer um índice que indique a posição onde um determinado valor será armazenado no vetor.

Por exemplo...

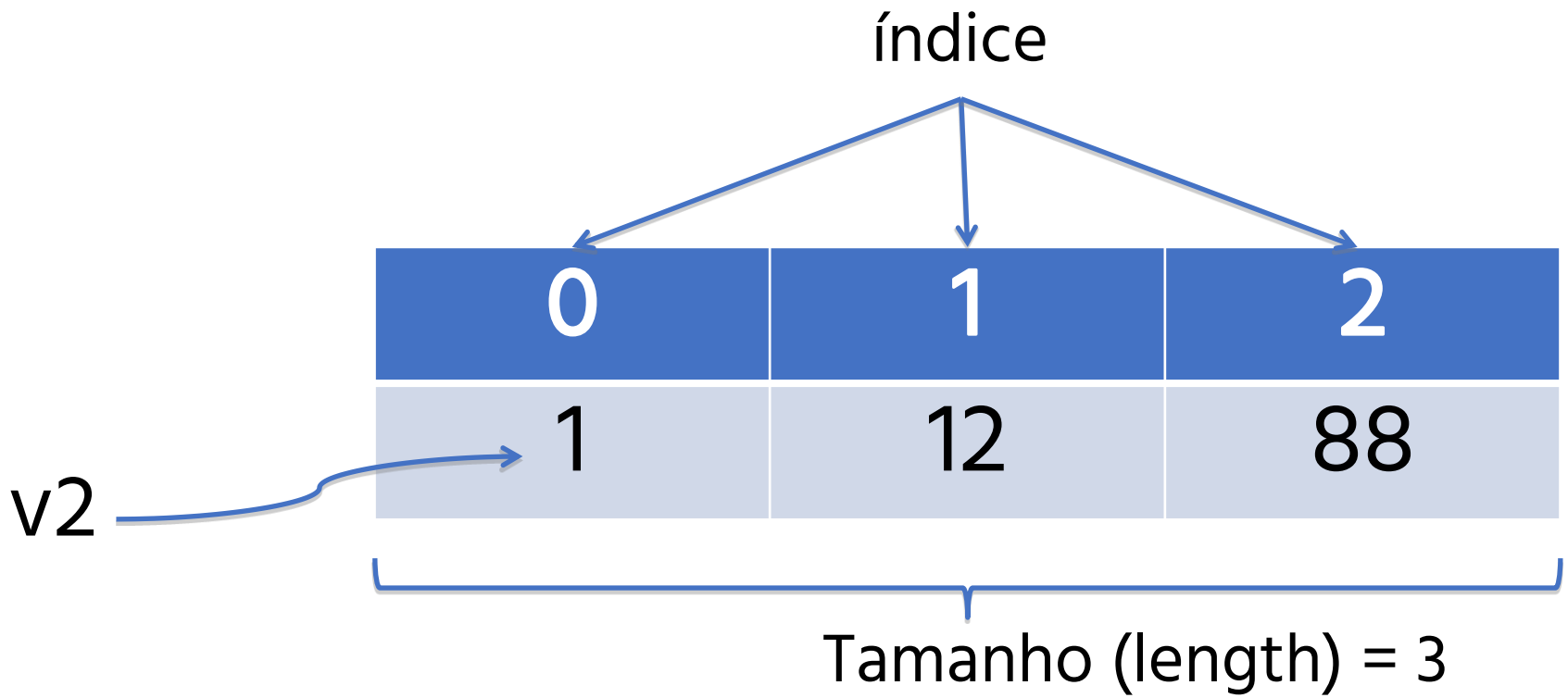
```
String cores[] = new String[3];

cores[0] = “vermelho";
cores[1] = “amarelo";
cores[2] = “azul";

System.out.println("Posição 0 do vetor: " + cores[0]);
System.out.println("Posição 1 do vetor: " + cores[1]);
System.out.println("Posição 2 do vetor: " + cores[2]);
```

Pergunta...


```
int[] v2= new int[3];
v2[0] = 1;
v2[1] = 12;
v2[2] = 88;

System.out.println(v2[2]);
System.out.println(v2[1]);
System.out.println(v2[0]);
```



Visualizando...



Acessando o primeiro e o último índice

- Os valores podem ser recuperados do vetor usando um índice (index).
- O primeiro valor está no índice zero e o último no tamanho do vetor menos 1 (**length – 1**).

```
int [] vetor = new int[20];
vetor[0] = 1;            // primeira posição do vetor.
vetor[vetor.length-1] = 19;        //última posição do vetor.
```

Inicializando o vetor com vários valores

```
int[] vetor = new int[3];
vetor[0] = 2;
vetor[1] = 8;
vetor[2] = 12;
```

...ou apenas...

```
int[] vetor = {2,8,12};
```

Preenchendo e mostrando os elementos de um vetor com for

- Para preencher um vetor, temos que atribuir valores para cada uma de suas posições.
- Para acessar todos os valores contidos em um vetor, temos que **percorrer todas as posições** do vetor e **obter cada valor** correspondente.

Preenchendo e mostrando os elementos de um vetor com for

- Deve-se implementar um **mecanismo para controlar o valor do índice**.
- Para percorrer um vetor, a estrutura de repetição for se apresenta como um bom recurso!
- A estrutura for permite contar um valor inicial até um valor final, que deve coincidir com os índices do vetor que está percorrendo.

Acessando o vetor com for tradicional

```
int [] vetor = {2,8,12};

for (int i = 0; i < vetor.length; i++) {
    System.out.println(vetor[i]);
}
```

for x foreach

Exibindo com o for each

- O for each itera por elementos de uma coleção (também funciona com matrizes e Strings).
- Por exemplo:

```
int [] vetor = {2,8,12};
for (int elemento : vetor) {
        System.out.println(elemento);
}
```

Estrutura do for each


```
for (int elemento: vetor) {
        System.out.println(elemento);
}
```

O elemento irá percorrer todos os itens da lista vetor. Em cada iteração o elemento receberá o valor de um item que está armazenado na lista. O elemento percorre desde a posição inicial até a final da lista.

Exibindo com o for each

O for each também funciona em vetores de objetos!

```
String[] nomes = {"Eric", "Elion", "Ethan"};
for(String nome : nomes) {
        System.out.println(nome);
}
```

Acessando o array com while

```
int [] vetor = {2,8,12};
int i = 0;
while (i < 3) {
    System.out.println(vetor[i]);
    i++;
}
```

Utilizando o .length no while

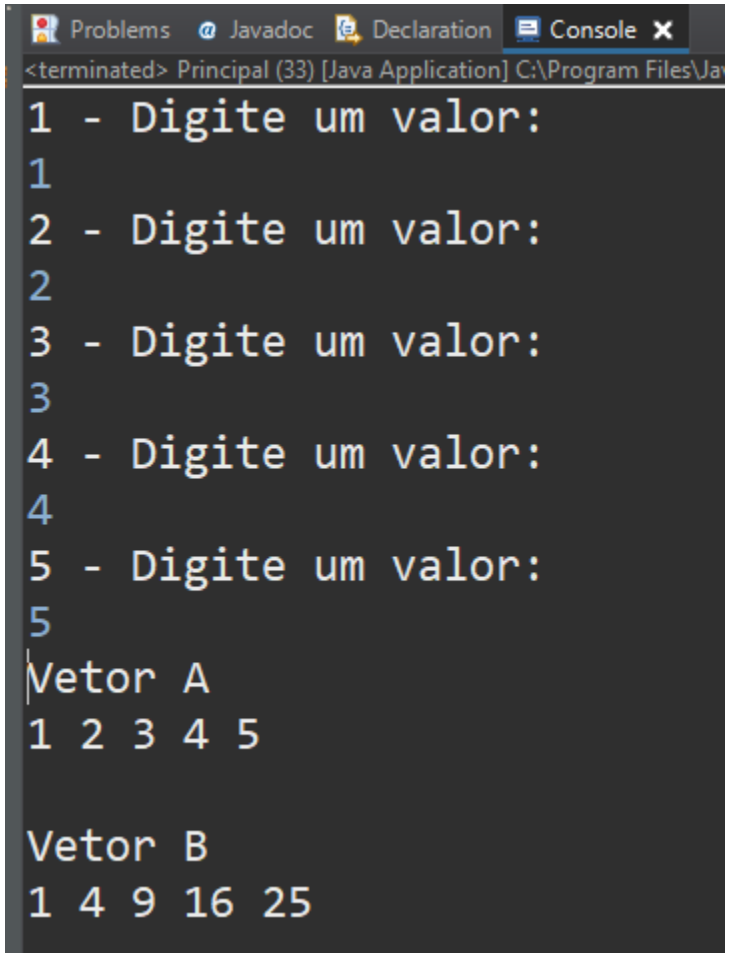
```
int [] vetor = {2,8,12};
int i = 0;
while (i < vetor.length) {
    System.out.println(vetor[i]);
    i++;
}
```

Alguns Possíveis Problemas...

```
int [] vetor1 = {2,8,12};
int [] vetor2 = new int[3];

System.out.println(vetor1[3]); // IndexOutOfBoundException

System.out.println(vetor1); // Mostra o endereço de memória

System.out.println(vetor2[0]); // Zero! O vetor de inteiros é iniciado com zeros
```

Exercício 1

- Crie um array (vetor) A com 5 elementos inteiros. Receba do usuário os 5 números e atribua a cada posição do vetor.
- Em seguida, construa um vetor B de mesmo tipo e tamanho, sendo que, cada elemento do vetor B deverá ser o quadrado do respectivo elemento do vetor A, ou seja:
- B[i] = A[i] * A[i]
- Mostre o resultado dos 2 vetores!



Resposta 1

*Principal.java

```java
import java.util.Scanner;

public class Principal {
    public static void main(String[] args) {
        int[] a = new int[5];
        int[] b = new int[5];
        Scanner teclado = new Scanner(System.in);
        for (int i=0; i<a.length; i++) {
            System.out.println((i+1)+" - Digite um valor: ");
            int valor = teclado.nextInt();
            a[i] = valor;
            b[i] = a[i] * a[i];
        }
        System.out.println("Vetor A");
        for (int i: a) {
            System.out.print(i + " ");
        }
        System.out.println("\n\nVetor B");
        for (int i: b) {
            System.out.print(i + " ");
        }
    }
}
```

Problems | Javadoc | Declaration | Console

<terminated> Principal (33) [Java Application] C:\Program Files\Ja

```
1 - Digite um valor:
1
2 - Digite um valor:
2
3 - Digite um valor:
3
4 - Digite um valor:
4
5 - Digite um valor:
5
Vetor A
1 2 3 4 5

Vetor B
1 4 9 16 25
```

Exercício 2

Faça um programa que receba 5 números e os armazena em um vetor X de 5 posições.

O programa deverá armazenar num vetor Y (também de 5 posições) a metade do valor digitado pelo usuário em cada posição do vetor X.

No final, o programa deverá escrever todo o conteúdo dos vetores X e Y na tela.

Problems Javadoc Declaration Console

<terminated> Principal (33) [Java Application] C:\Program Files\

```
1 - Digite um valor:
2
2 - Digite um valor:
4
3 - Digite um valor:
6
4 - Digite um valor:
8
5 - Digite um valor:
10
Vetor X
2 4 6 8 10

Vetor Y
1 2 3 4 5
```

Resposta 2

```java
import java.util.Scanner;

public class Principal {
    public static void main(String[] args) {
        int[] x = new int[5];
        int[] y = new int[5];
        Scanner teclado = new Scanner(System.in);
        for (int i=0; i<x.length; i++) {
            System.out.println((i+1)+" - Digite um valor: ");
            int valor = teclado.nextInt();
            x[i] = valor;
            y[i] = x[i] /2;
        }
        System.out.println("Vetor X");
        for (int i: x) {
            System.out.print(i + " ");
        }
        System.out.println("\n\nVetor Y");
        for (int i: y) {
            System.out.print(i + " ");
        }
    }
}
```

```
Problems  Javadoc  Declaration  Console
<terminated> Principal (33) [Java Application] C:\Program Files\
1 - Digite um valor:
2
2 - Digite um valor:
4
3 - Digite um valor:
6
4 - Digite um valor:
8
5 - Digite um valor:
10
Vetor X
2 4 6 8 10

Vetor Y
1 2 3 4 5
```

Exercício Extra!

Elabore um programa para ler 10 valores.

Caso o valor digitado seja par, armazenar no vetor **par**.

Caso seja ímpar, armazenar no vetor **ímpar**.

O tamanho de cada vetor deve ser de 10 posições.

Ao final, mostre o conteúdo dos 2 vetores.

```
Console  Problems
<terminated> Principal [Java A
Digite o 1º número:
1
Digite o 2º número:
2
Digite o 3º número:
3
Digite o 4º número:
4
Digite o 5º número:
5
Digite o 6º número:
6
Digite o 7º número:
7
Digite o 8º número:
8
Digite o 9º número:
9
Digite o 10º número:
10
Valores do Vetor Par
2
4
6
8
10
Valores do Vetor Ímpar
1
3
5
7
9
```

Exercício Extra - Resposta

```java
import java.util.Scanner;

public class Principal {
    public static void main (String[] args) {
        int[] par = new int[10];
        int[] ímpar = new int[10];
        int indimpar = 0;
        int indpar = 0;

        Scanner teclado = new Scanner(System.in);
        for (int i=1; i<=10; i++) {
            System.out.println("Digite o "+i+"º número:");
            int resposta = teclado.nextInt();
            if (resposta%2==0) {
                par[indpar] = resposta;
                indpar++;
            } else {
                ímpar[indimpar] = resposta;
                indimpar++;
            }
        }

        System.out.println("Valores do Vetor Par");
        for (int i=0; i<indpar; i++){
            System.out.println(par[i]);
        }
        System.out.println("Valores do Vetor Ímpar");
        for (int i=0; i<indimpar; i++){
            System.out.println(ímpar[i]);
        }
    }
}
```

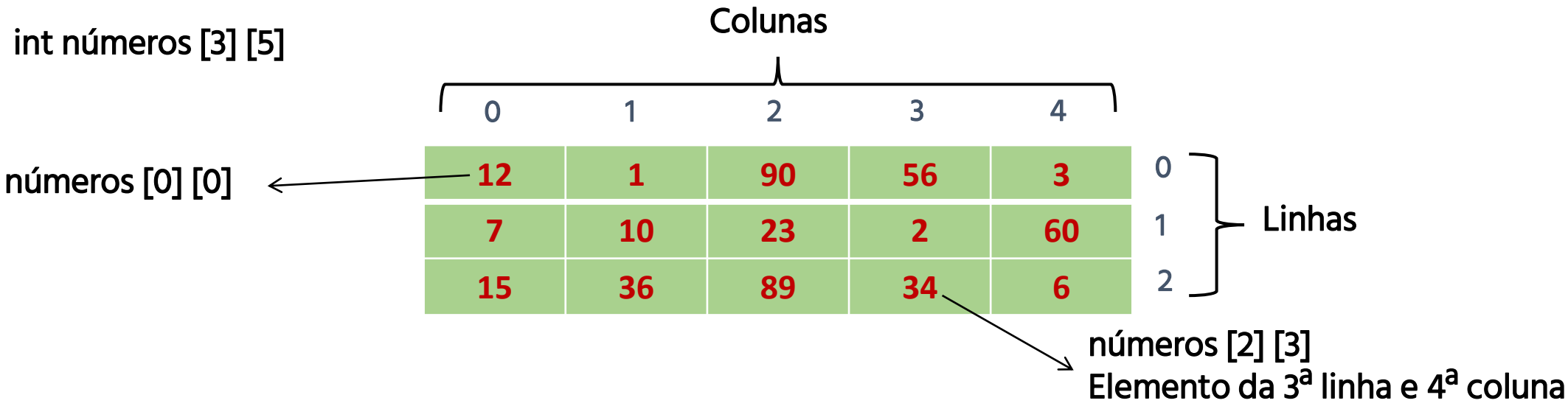
```
Console  Problems
<terminated> Principal [Java A
Digite o 1º número:
1
Digite o 2º número:
2
Digite o 3º número:
3
Digite o 4º número:
4
Digite o 5º número:
5
Digite o 6º número:
6
Digite o 7º número:
7
Digite o 8º número:
8
Digite o 9º número:
9
Digite o 10º número:
10
Valores do Vetor Par
2
4
6
8
10
Valores do Vetor Ímpar
1
3
5
7
9
```

Matrizes

O que é uma Matriz?

- É um conjunto de **variáveis multidimensionais, de um mesmo tipo**, que possuem o **mesmo identificador** (nome) e são **alocadas sequencialmente na memória**.
- Também conhecida como variável homogênea multidimensional.
- Uma variável do tipo matriz precisa de um índice para cada uma de suas dimensões.

Exemplo de uma Matriz



números: é o nome da matriz bidimensional, onde o tamanho a 1ª dimensão (linha) é 3 e o tamanho da 2ª dimensão (coluna) é 5.

Em vermelho: são os elementos da matriz.

Em azul: são os índices, ou seja, a posição dos elementos na matriz (linha, coluna).

Matrizes

- Exemplo de declaração de uma matriz:

```
int  matriz[][] = new int[5][5];
```

ou

```
int[][]  matriz = new int[5][5];
```

- Os dois conjuntos de colchetes indicam a declaração de uma matriz.
- É preciso definir o tamanho da matriz ([5][5]) e alocar memória para o armazenamento dos elementos (new int).
- O primeiro valor representa a quantidade de linhas, o segundo, a quantidade de colunas.

Comparando a instanciação de Vetores e Matrizes

- Vetor

```
int vetor[] = new int[10];
```

- Matriz

```
int matriz[][] = new int[5][5];
```

Como atribuir valores a uma matriz?

Devemos fornecer um índice que indique a posição que um determinado valor será armazenado no vetor. Por exemplo...

```
int tabela[][] = new int[2][2];

tabela[0][0] = 1;
tabela[0][1] = 5;
tabela[1][0] = 7;
tabela[1][1] = 2;

System.out.println(“Linha 0, Coluna 0: " + tabela[0][0]);
System.out.println("Linha 1, Coluna 0: " + tabela[1][0]);
System.out.println("Linha 1, Coluna 1: " + tabela[1][1]);
```

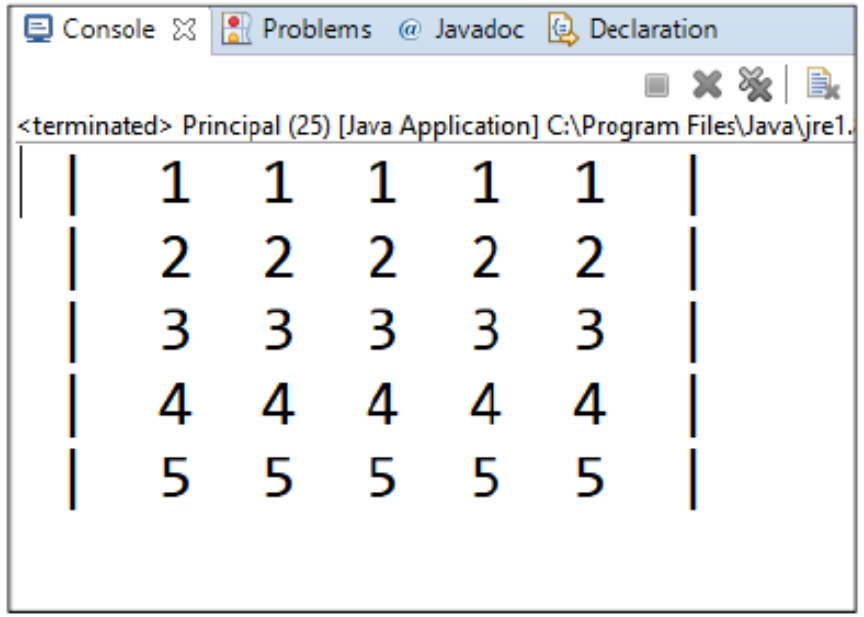
Como atribuir e escrever os valores de uma maneira mais eficiente?

- Deve-se implementar um mecanismo que controle o valor dos índices.
- Para percorrer uma matriz a estrutura de repetição **for** se apresenta como um bom recurso.
- A estrutura **for** permite contar um valor inicial até um valor final, coincidentemente com os índices de uma matrizes.
- No entanto, precisaremos utilizar duas estruturas **for**! Uma para iterar as linhas e outra para iterar nas colunas.

Acessando elementos da matriz com for

```
int matriz [][] = new int[3][3];


for (int linha = 0; linha <3; linha ++) {
        for (int coluna = 0; coluna <3; coluna++) {
                System.out.println(matriz[linha][coluna]);
        }
}
```

Exercício 3



E se eu quiser exibir a matriz como uma matriz?

5 minutos

Façam uma matriz igual a essa sem interagir com o usuário.

Resposta 3

*Principal.java

```
1
2  public class Principal {
3
4      public static void main(String[] args) {
5          int[][] matriz = new int[5][5];
6
7          for (int linha=0; linha < 5; linha ++) {
8              for (int coluna=0; coluna <5; coluna++) {
9                  matriz[linha][coluna] = linha+1;
10             }
11         }
12
13         for (int linha=0; linha < 5; linha ++) {
14             System.out.print(" | ");
15             for (int coluna=0; coluna <5; coluna++) {
16                 System.out.print(" "+matriz[linha][coluna]+" ");
17             }
18             System.out.println(" | ");
19         }
20     }
21 }
```

Console | Problems | Javadoc | Declaration

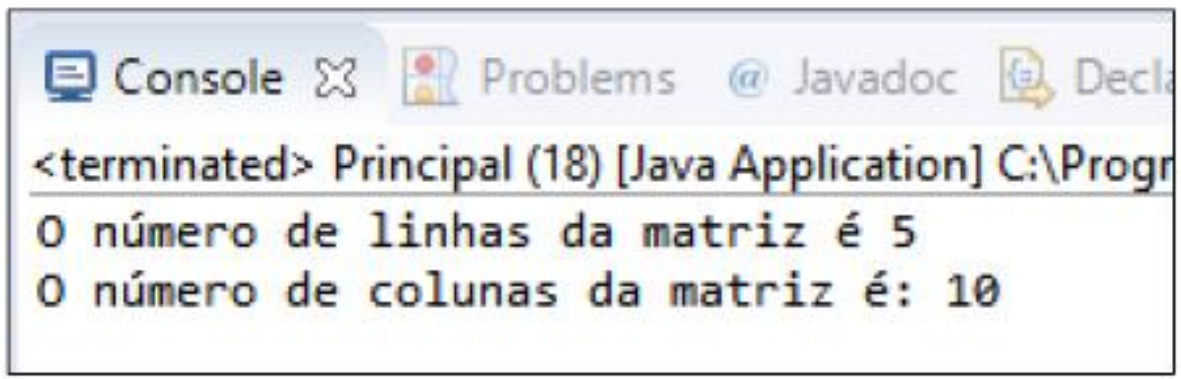
<terminated> Principal (25) [Java Application] C:\Program Files\Java\jre1.

```
|  1  1  1  1  1  |
|  2  2  2  2  2  |
|  3  3  3  3  3  |
|  4  4  4  4  4  |
|  5  5  5  5  5  |
```

Como descobrimos a quantidade de linhas e colunas de uma matriz?

Descobrindo a quantidade de linhas e colunas de uma matriz

```java
int[][] matriz = new int[5][10];
System.out.println("O número de linhas da matriz é "+matriz.length);
System.out.println("O número de colunas da matriz é: "+matriz[0].length);
```



Exercício 4

Faça um programa em Java que recebe do usuário 9 números inteiros e armazena numa matriz 3x3. Em seguida, o programa deverá escrever cada número e sua posição na matriz.

```
import java.util.Scanner;
public class tarefa {
        public static void main(String[] args) {
        Scanner entrada = new Scanner(System.in);
            int num[][] = new int[3][3];

             for(int linha=0; linha<3; linha++) {
                    for(int coluna=0; coluna<3; coluna++) {
                        System.out.println("Digite um número: ");
                        num[linha][coluna] = entrada.nextInt();
                    }
               }

            for(int linha=0; linha<3; linha++) {
                    for(int coluna=0; coluna<3; coluna++) {
                        System.out.println(linha + "-" + coluna + ": " +
                        num[linha][coluna]);
                    }
               }
        }
}
```

Resposta 4

```java
import java.util.Scanner;

public class Principal {
    public static void main (String[] args) {
        int[][] matriz = new int[3][3];
        Scanner teclado = new Scanner(System.in);
        //receber valores do usuário
        for (int linha=0; linha<3; linha++) {
            for (int coluna = 0; coluna<3; coluna++) {
                System.out.println("digite o elemento matriz["+linha+"]["+coluna+"]");
                matriz[linha][coluna] = teclado.nextInt();
            }
        }
        //exibindo os valores
        for (int linha=0; linha<3; linha++) {
            System.out.print(" | ");
            for (int coluna = 0; coluna<3; coluna++) {
                System.out.print(matriz[linha][coluna]+" ");
            }
            System.out.println(" | ");
        }
    }
}
```

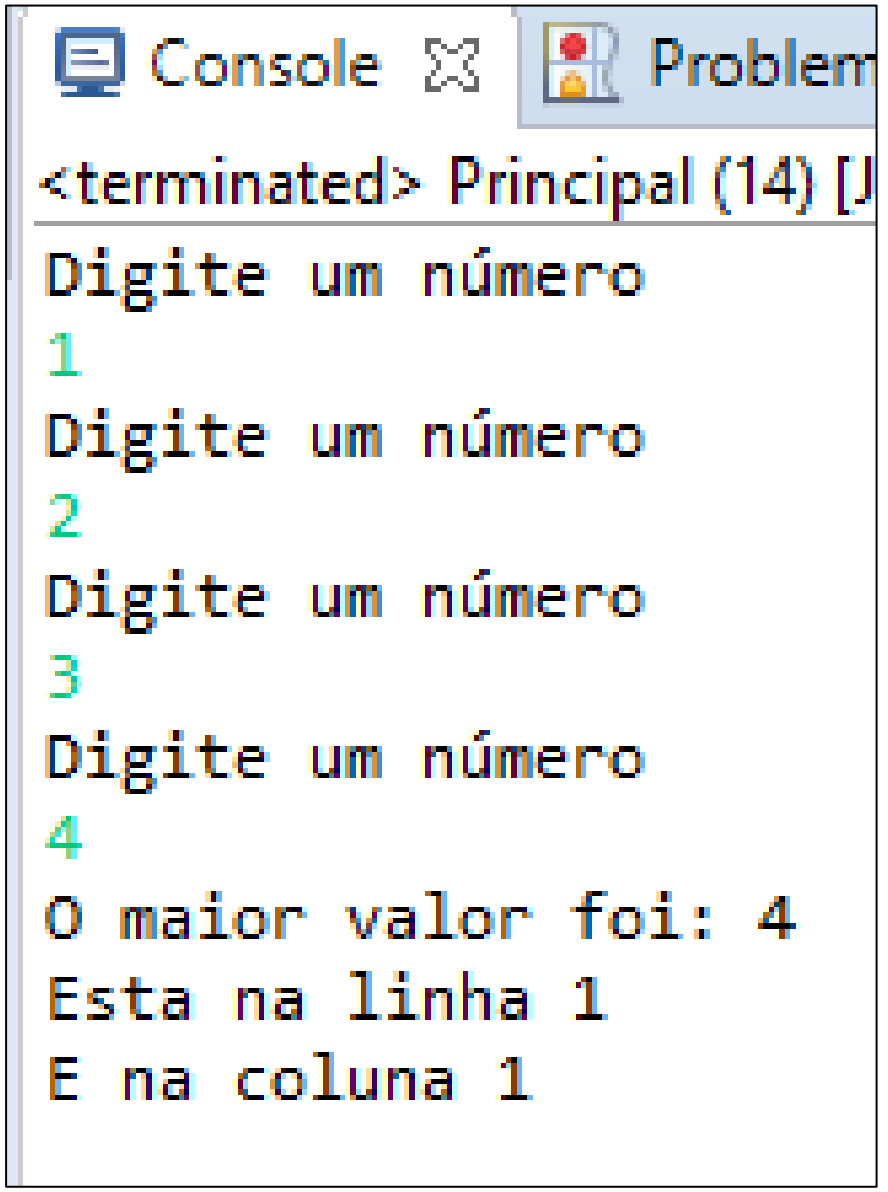
Console Problems @ Javad

<terminated> Principal [Java Application]

```
digite o elemento matriz[0][0]
1
digite o elemento matriz[0][1]
2
digite o elemento matriz[0][2]
3
digite o elemento matriz[1][0]
4
digite o elemento matriz[1][1]
5
digite o elemento matriz[1][2]
6
digite o elemento matriz[2][0]
7
digite o elemento matriz[2][1]
8
digite o elemento matriz[2][2]
9
 | 1 2 3  |
 | 4 5 6  |
 | 7 8 9  |
```

Exercício 5

Faça um programa para ler valores inteiros em uma matriz (2,2). O programa deverá encontrar o maior valor contido na matriz e apresentar sua posição.



Resposta Exercício 5

*Principal.java

```java
import java.util.Scanner;

public class Principal {

    public static void main(String[] args) {
        int matriz[][] = new int[2][2];
        Scanner teclado = new Scanner(System.in);
        int maior = 0;
        int lm = 0;
        int cm = 0;
        for (int linha=0; linha<matriz.length; linha++) {
            for (int coluna=0; coluna<matriz[0].length; coluna++) {
                System.out.println("Digite um número");
                matriz[linha][coluna] = teclado.nextInt();
                if (matriz[linha][coluna]>maior) {
                    maior = matriz[linha][coluna];
                    lm = linha;
                    cm = coluna;
                }
            }
        }
        System.out.println("O maior valor foi: "+maior);
        System.out.println("Esta na linha "+lm);
        System.out.println("E na coluna "+cm);
        teclado.close();
    }
}
```

Console | Problem

<terminated> Principal (14) [J

```
Digite um número
1
Digite um número
2
Digite um número
3
Digite um número
4
O maior valor foi: 4
Esta na linha 1
E na coluna 1
```

Intervalo: 15 minutos

Lista Sequencial
x
Lista Encadeada

Lista Sequencial

Estrutura que organiza a informação em índices.

- **Vantagem**: Pode acessar qualquer elemento com seu índice.
- **Desvantagem**: Não é possível adicionar ou remover elementos (tamanho da lista é estático).

Lista Encadeada

É uma estrutura de dados na qual um elemento da lista é um nó que referencia o próximo elemento da lista.

Cada nó é composto de alguns dados e uma referência para outro nó.

- **Vantagem**: Pode adicionar e remover elementos (tamanho dinâmico).
- **Desvantagem**: Tem que atravessar a lista do começo para encontrar um elemento.

Como aproveitar o melhor das 2 estruturas?

Como aproveitar o melhor das 2 estruturas?

- Lista sequenciais são fracas onde listas encadeadas são fortes (e vice versa). Existe alguma estrutura que tira vantagem disso? SIM!!

- A API Java provê uma classe chamada **ArrayList**.

- Um ArrayList é uma estrutura de dados que é uma combinação de uma **lista sequencial** com uma **lista encadeada**.

A Classe ArrayList

- O ArrayList contém elementos que podem ser acessados por seus índices.
- Principais vantagens sobre as listas sequenciais:
  1. Um ArrayList **expande "automaticamente"** à medida em que os itens são adicionados nele.
  2. Um ArrayList **encolhe "automaticamente"** quando itens são removidos dele.

- Para usar ArrayList você deve importá-lo do pacote java.util: **import java.util.ArrayList;**

Implementação

Crie um projeto com
a classe Principal e
adicione o método main.

A Classe ArrayList

- A declaração e criação de um ArrayList segue o formato:

  ArrayList<dataType> identificador = new ArrayList<dataType>();

- Então, para criar um ArrayList do tipo String:

  ArrayList<String> **nomes** = new ArrayList<String>();

Importante

- O ArrayList pode manter apenas elementos que são **objetos**.

- Isso significa que o ArrayList não consegue manter elementos de tipos primitivos.
- **E agora????**

Agora a gente utiliza as classes Wrapper

São classes que empacotam os tipos primitivos de dados.

ArrayList de “tipos primitivos”

- Existem classes que representam os tipos primitivos.
- Classe **Integer** guarda o tipo **int**.
- Classe **Double** guarda o tipo **double**...

Forma errada:

~~ArrayList<**int**> números = new ArrayList<**int**>();~~

Forma correta:

ArrayList<**Integer**> números = new ArrayList<**Integer**>();

Métodos de Manipulação do ArrayList

1. **add**(arg1) e **add**(arg1, arg2)
2. **remove**(arg)
3. **set**(arg1, arg2)
4. **get**(arg)
5. **size**()

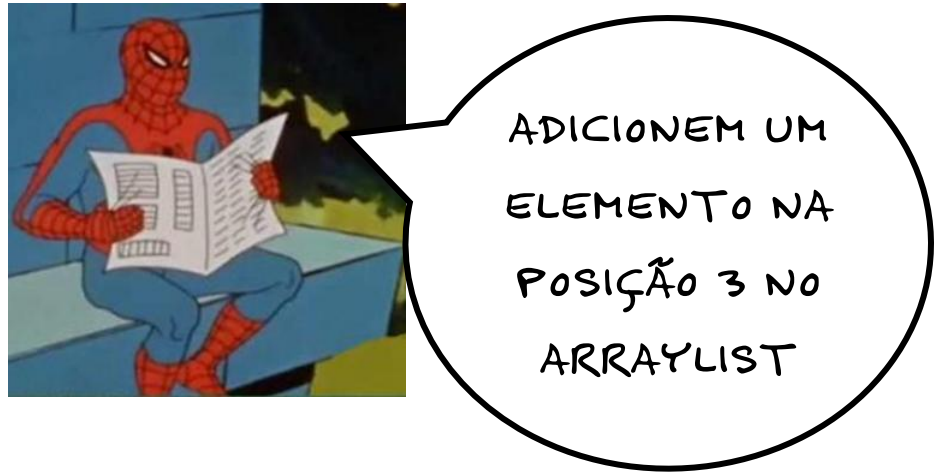
Adicionando Elementos Método add()

- Você pode adicionar itens ao final de um ArrayList usando o método **add**.

```
nomes.add(“Ana");        //posição 0
nomes.add(“Bruno");      //posição 1
nomes.add(“Carlos");     //posição 2
```

- Isto irá tornar ”Ana” o primeiro elemento, “Bruno” o segundo elemento e “Carlos” o terceiro elemento.
- Assim como num vetor, esses elementos tem índices que começam em zero.

Adicionando Elementos em Posição Específica



- Para inserir um elemento num índice específico, use o método **add** com **dois argumentos**: a posição e o objeto.

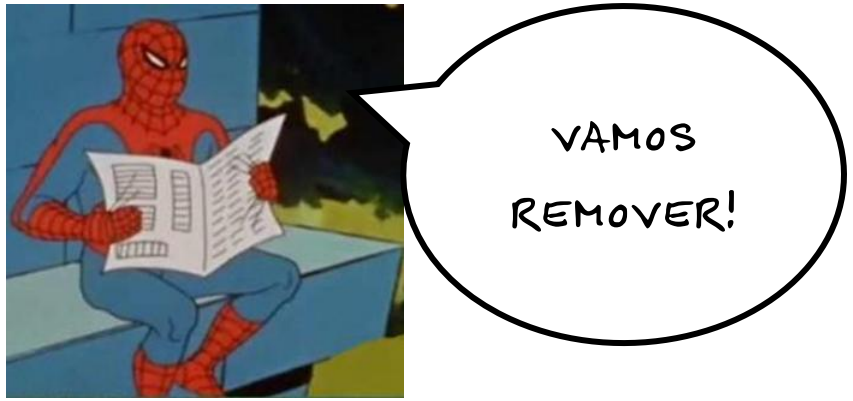
```
nomes.add(3, "Daniel");
```

- Isto irá adicionar "Daniel" ao índice 3 da lista e todos os elementos com índice maior que dois, irão aumentar um índice.
- Então, se um elemento ocupa o índice 5 antes de adicionarmos o elemento no índice 3, após adicioná-lo, ele será movido para o índice 6.

Exceção!

- E se eu tentar adicionar numa posição maior que o tamanho do ArrayList?
- **Será disparada a exceção index.outOfBoundException!**

Removendo elementos método remove()

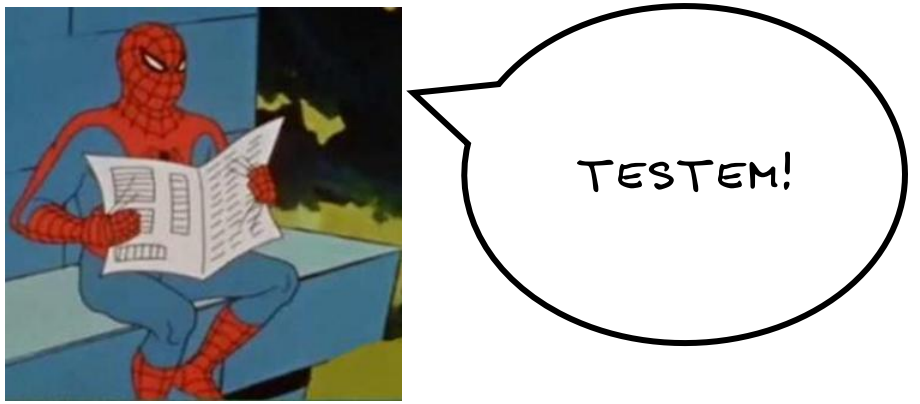


- Para remover um elemento da lista, use o método **remove** com o índice do elemento que você quer remover:

```
nomes.remove(1);
```

- Isto irá remover o segundo elemento da lista e todos os elementos que vêm depois do índice 1, se moverão para o índice anterior.
- Então, se um elemento ocupa o índice 5 antes da remoção do elemento de índice 1, após a remoção, este elemento ocupará índice 4.

Trocando Elementos
Método set()



- Você pode trocar um elemento usando o método set:

```
nomes.set(2, “Carol");
```

- Isto irá **trocar** o valor do elemento do índice 2 para o valor “Carol“.
- Irá **deletar** o elemento que estiver na posição 2 e adicionar “Carol".

Exibindo elementos
Método get()

- Você pode acessar o conteúdo do ArrayList com o método get.
- É preciso inserir o índice do elemento que deseja acessar:

```
String nome = nomes.get(2);
System.out.println(nome);
System.out.println(nomes.get(1));
```

Descobrindo o tamanho do ArrayList. Método size()

- Para descobrir o número de objetos armazenados num ArrayList, use o método **size**.
- O retorno é igual ao membro de classe length de um vetor.

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

Resumo do código (até agora)

```
public class Principal {
        public static void main (String[] args) {
                ArrayList<String> nomes = new ArrayList<String>();
                nomes.add(“Ana");
                nomes.add(“Bruno");
                nomes.add(“Carlos");
                nomes.add(3, “Daniel”);
                nomes.remove(1);
                nomes.set(2, “Carol");
                System.out.println(nomes.size());
        }
}
```

E como podemos exibir o conteúdo do ArrayList?

```
public class Principal {
        public static void main (String[] args) {
                ArrayList<String> nomes = new ArrayList<String>();
                        (...)
                for (String n: nomes) System.out.println(n);
                }
}
```

Coleções

- Arraylist é apenas um tipo de coleção que podemos utilizar em Java.
- Existem outros tipos de coleções que podem ser utilizadas.

Qual o resultado?


```
                                    //Teste de Mesa
ArrayList<String> amigos = new ArrayList<String>();
amigos.add("joao");          // joao
amigos.add("maria");         // joao, maria
amigos.add("ana");           // joao, maria, ana
amigos.remove(1);            // joao, ana
amigos.remove(1);            // joao
```

Material de Apoio

- Aula do professor sobre ArrayList (40 minutos)
- https://youtu.be/l0hSoVpmqpA

Lista de Exercícios 6

Mestre Jedi

PROGRAMAÇÃO ORIENTADA A OBJETOS

AULA 8 – VETORES, MATRIZES E ARRAYLIST

WALTER TRAVASSOS SARINHO

@PROF.WALTERTRAVASSOS

WALTER.TRAVASSOS@SECTRAS.EDU.BR