

ReactJS



spring boot



Prof.º

Alexandre Gomes

>alexandre.silva251@fatec.sp.gov.br

>(16) 99201-1010



"Java"

# Hello word em JAVA

```
package com.mycompany.fundamentos;
```

```
/**
```

```
 *
```

```
 * @author Alexandre
```

```
 */
```

```
public class Fundamentos {
```

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

```
        System.out.println("Hello World!");
```

```
    }
```

```
}
```

# Palavras-Chaves do JAVA

- O Java possui 53 palavras-chaves e palavras reservadas:

abstract	class	extends	implements	null	strictfp	true
assert	const	false	import	package	super	try
boolean	continue	final	instanceof	private	switch	void
break	default	finally	int	protected	synchronized	volatile
byte	do	float	interface	public	this	while
case	double	for	long	return	throw	
catch	else	goto	native	short	throws	
char	enum	if	new	static	transient	

# Algumas sequências de escape comuns.

Sequência de escape	Descrição
<code>\n</code>	Nova linha. Posiciona o cursor de tela no início da <i>próxima</i> linha.
<code>\t</code>	Tabulação horizontal. Move o cursor de tela para a próxima parada de tabulação.
<code>\r</code>	Retorno de carro. Posiciona o cursor da tela no início da linha <i>atual</i> — <i>não</i> avança para a próxima linha. Qualquer saída de caracteres depois do retorno de carro <i>sobrescreve</i> a saída de caracteres anteriormente gerada na linha atual.
<code>\\</code>	Barras invertidas. Utilizadas para imprimir um caractere de barra invertida.
<code>\"</code>	Aspas duplas. Utilizadas para imprimir um caractere de aspas duplas. Por exemplo, <pre>System.out.println("\entre aspas\");</pre> exibe "entre aspas".

# Aritmética

Operação Java	Operador	Expressão algébrica	Expressão Java
Adição	+	$f + 7$	<code>f + 7</code>
Subtração	-	$p - c$	<code>p - c</code>
Multiplicação	*	$bm$	<code>b * m</code>
Divisão	/	$x/y$ ou $\frac{x}{y}$ ou $x \div y$	<code>x / y</code>
Resto	%	$r \bmod s$	<code>r % s</code>

Operador(es)	Operação(ões)	Ordem de avaliação (precedência)
* / %	Multiplicação Divisão Resto	Avaliado primeiro. Se houver vários operadores desse tipo, eles são avaliados da <i>esquerda para a direita</i> .
+ -	Adição Subtração	Avaliado em seguida. Se houver vários operadores desse tipo, eles são avaliados da <i>esquerda para a direita</i> .
=	Atribuição	Avaliado por último.

### *Exemplo de expressões algébricas e Java*

Agora vamos considerar várias expressões à luz das regras de precedência de operador. Cada exemplo lista uma expressão algébrica e seu equivalente Java. O seguinte é de uma média aritmética dos cinco termos:

*Álgebra:*  $m = \frac{a + b + c + d + e}{5}$

*Java:* `m = (a + b + c + d + e) / 5;`

*Álgebra:*  $z = pr \% q + w / x - y$

*Java:* `z = p * r % q + w / x - y;`

6

1

2

4

3

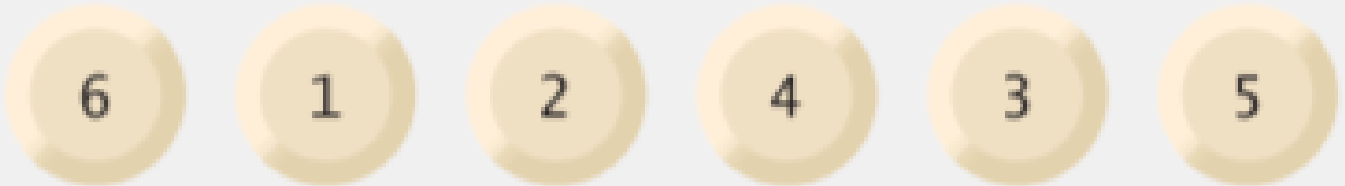
5

### *Avaliação de um polinômio de segundo grau*

Para desenvolver um bom entendimento das regras de precedência de operadores, considere a avaliação de uma expressão de atribuição que inclui um polinômio de segundo grau  $ax^2 + bx + c$ :

### *Avaliação de um polinômio de segundo grau*

Para desenvolver um bom entendimento das regras de precedência de operadores, considere a avaliação de uma expressão de atribuição que inclui um polinômio de segundo grau  $ax^2 + bx + c$ :

$$y = a * x * x + b * x + c;$$


The diagram illustrates the evaluation order of the expression  $y = a * x * x + b * x + c;$  using numbered yellow circles. The numbers are placed below the tokens in the expression as follows:

- 6 is below  $a$
- 1 is below the first  $*$
- 2 is below the second  $*$
- 4 is below the first  $x$  (the one after the second  $*$ )
- 3 is below the  $*$  between  $b$  and  $x$
- 5 is below the  $x$  after the  $*$  between  $b$  and  $x$

The expression is:  $y = a * x * x + b * x + c;$



$$y = (a * x * x) + (b * x) + c;$$

Passo 1

$$y = 2 * 5 * 5 + 3 * 5 + 7; \quad (\text{Multiplicação mais à esquerda})$$

$$2 * 5 \text{ é } 10$$

Passo 2

$$y = 10 * 5 + 3 * 5 + 7; \quad (\text{Multiplicação mais à esquerda})$$

$$10 * 5 \text{ é } 50$$

Passo 3

$$y = 50 + 3 * 5 + 7; \quad (\text{Multiplicação antes da adição})$$

$$3 * 5 \text{ é } 15$$

Passo 4

$$y = 50 + 15 + 7; \quad (\text{Adição mais à esquerda})$$

$$50 + 15 \text{ é } 65$$

Passo 5

$$y = 65 + 7; \quad (\text{Última adição})$$

$$65 + 7 \text{ é } 72$$

Passo 6

$$y = 72 \quad (\text{Última operação — coloca 72 em y})$$

# SOMA

```
package com.mycompany.fundamentos;
import java.util.Scanner;

/**
 *
 * @author Alexandre
 */
public class Soma {
    public static void main(String[] args)
    {
        //cria um Scanner para obter entrada a partir da janela de comando
        Scanner input = new Scanner(System.in);
        int number1, number2, sum;

        System.out.print("Digite o primeiro numero: ");
        number1 = input.nextInt();

        System.out.print("Digite o segundo numero: ");
        number2 = input.nextInt();

        sum = number1 + number2;

        System.out.printf("A Soma é %d%n", sum);
    }
}
```

# Tomada de decisão: operadores de igualdade e operadores relacionais

Operador algébrico	Operador de igualdade ou relacional Java	Exemplo de condição em Java	Significado da condição em Java
<i>Operadores de igualdade</i>			
=	==	x == y	x é igual a y
≠	!=	x != y	x é não igual a y
<i>Operadores relacionais</i>			
>	>	x > y	x é maior que y
<	<	x < y	x é menor que y
≥	>=	x >= y	x é maior que ou igual a y
≤	<=	x <= y	x é menor que ou igual a y

# COMPARAÇÕES

```
package com.mycompany.fundamentos;
import java.util.Scanner;
/**
 *
 * @author Alexandre
 */
public class Comparacao {
    public static void main(String[] args){
        Scanner input = new Scanner(System.in);
        int number1, number2;

        System.out.print("Digite o primeiro numero: ");
        number1 = input.nextInt();

        System.out.print("Digite o segundo numero: ");
        number2 = input.nextInt();

        if(number1 == number2)
            System.out.printf("%d == %d\n",number1, number2);

        if(number1 != number2)
            System.out.printf("%d != %d\n",number1, number2);

        if(number1 < number2)
            System.out.printf("%d < %d\n",number1, number2);

        if(number1 > number2)
            System.out.printf("%d > %d\n",number1, number2);

        if(number1 <= number2)
            System.out.printf("%d <= %d\n",number1, number2);

        if(number1 >= number2)
            System.out.printf("%d >= %d\n",number1, number2);
    }
}
```

# EXERCÍCIOS P/ ENTREGAR EM FOLHA

01 - Escreva em uma folha para entregar - Declarações, instruções ou comentários que realizem cada uma das tarefas a seguir:

- a) Crie um Scanner chamado input que leia valores a partir da entrada padrão.
- b) Declare as variáveis “a”, “b”, “c” e “resultado” como tipo inteiro.
- c) Solicite que o usuário insira o primeiro inteiro.
- d) Leia o primeiro inteiro digitado pelo usuário e armazene-o na variável a.
- e) Solicite que o usuário insira o segundo inteiro.
- f) Leia o segundo inteiro digitado pelo usuário e armazene-o na variável b.
- g) Solicite que o usuário insira o terceiro inteiro.
- h) Leia o terceiro inteiro digitado pelo usuário e armazene-o na variável c.
- i) Compute o produto dos três inteiros contidos nas variáveis “a”, “b” e “c” e atribua o resultado à variável “resultado”.
- j) Use System.out.printf para exibir a mensagem “O produto é ” seguida pelo valor da variável “resultado”.

# EXERCÍCIOS P/ ENTREGAR EM FOLHA

a) `Scanner input = new Scanner(System.in);`

b) `int a, b, c, resultado;`

ou

`int a;`

`int b;`

`int c;`

`int resultado;`

c) `System.out.print("Entre com o primeiro inteiro: ");`

d) `a = input.nextInt();`

e) `System.out.print("Entre com o segundo inteiro: ");`

f) `b = input.nextInt();`

g) `System.out.print("Entre com o terceiro inteiro: ");`

h) `c = input.nextInt();`

i) `resultado = a * b * c;`

j) `System.out.printf("O Produto é %d%n", resultado);`

# EXERCÍCIOS P/ ENTREGAR EM FOLHA

02 – Agora, usando as instruções que você escreveu no exercício 1, elabore um programa completo que calcule e imprima o produto de três inteiros.

```
1 // Exercício 2.6: Product.Java
2 // Calcula o produto de três inteiros.
3 import java.util.Scanner; // programa utiliza Scanner
4
5 public class Product
6 {
7     public static void main(String[] args)
8     {
9         // cria Scanner para obter entrada a partir da janela de comando
10         Scanner input = new Scanner(System.in);
11
12         int x; // primeiro número inserido pelo usuário
13         int y; // segundo número inserido pelo usuário
14         int z; // terceiro número inserido pelo usuário
15         int result; // produto dos números
16
17         System.out.print("Enter first integer: "); // solicita entrada
18         x = input.nextInt(); // lê o primeiro inteiro
19
20         System.out.print("Enter second integer: "); // solicita entrada
21         y = input.nextInt(); // lê o segundo inteiro
22
23         System.out.print("Enter third integer: "); // solicita entrada
24         z = input.nextInt(); // lê o terceiro inteiro
25
26         result = x * y * z; // calcula o produto dos números
27
28         System.out.printf("Product is %d\n", result);
29     } // fim do método main
30 } // fim da classe Product
```



03 – Escreva um aplicativo que exiba os números 1 a 4 na mesma linha, com cada par de adjacentes separados por um espaço. Use as seguintes técnicas:

- a) Uma instrução `System.out.println`.
- b) Quatro instruções `System.out.print`.
- c) Uma instrução `System.out.printf`.

```
package com.mycompany.fundamentos;
```

```
/**
```

```
 *
```

```
 * @author Alexandre
```

```
 */
```

```
public class Impressao {
```

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

```
        //a) Uma instrução System.out.println.  
        System.out.println("1 2 3 4");
```














```
        //b) Quatro instruções System.out.print  
        System.out.print("1 ");  
        System.out.print("2 ");  
        System.out.print("3 ");  
        System.out.print("4");
```

```
        //c) Uma instrução System.out.printf.  
        System.out.printf("%n%d %d %d %d", 1, 2, 3, 4);
```

```
    }
```

```
}
```

# EXEMPLO DA ESTRUTURA DE UM PROJETO REAL

- ▼  clinica
  - ▶  JRE System Library [JavaSE-12]
  - ▼  src
    - ▼  administracao
      - ▶  Agenda.java
    - ▼  pessoa
      - ▶  Funcionario.java
      - ▶  Medico.java
      - ▶  Paciente.java
    - ▼  tela
      - ▶  TelaAgendamenrto.java
      - ▶  TelaPrincipal.java
- ▶  exercicios

04 – CRIAR UMA ESTRUTURA DE UMA  
APLICAÇÃO PARA UMA ESCOLA OU  
FACULDADE (PACOTES / ARQUIVOS)

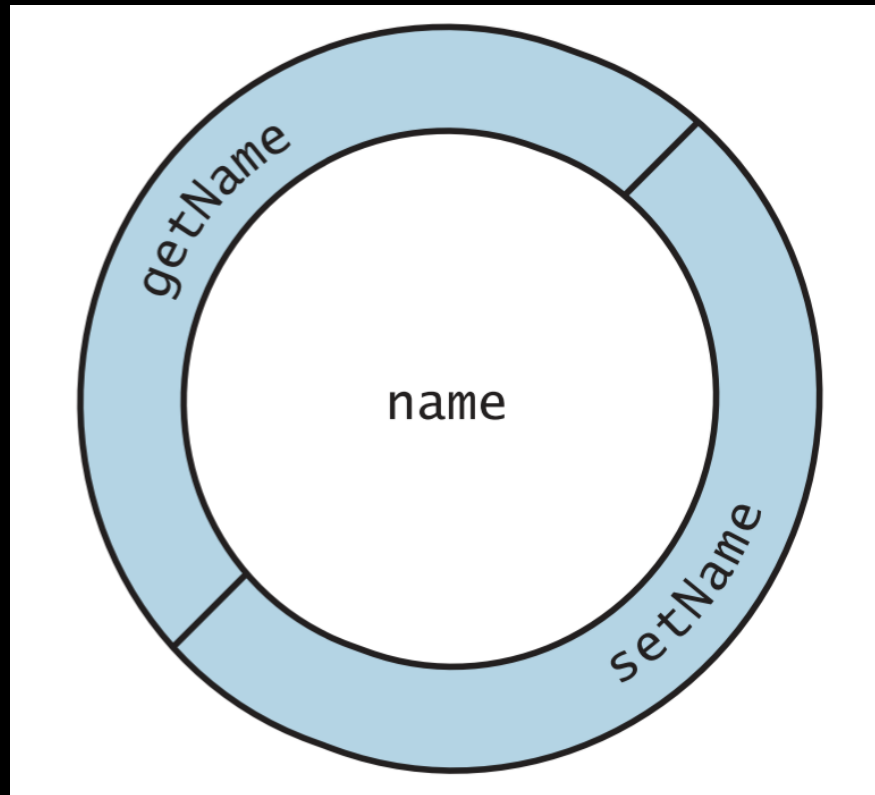
- faculdade
  - JRE System Library [JavaSE-12]
  - src
    - administracao
      - Agenda.java
      - Matricula.java
    - educacional
      - Disciplina.java
      - Prova.java
    - pessoa
      - Aluno.java
      - Funcionario.java
      - Professor.java

# Introdução a classes, objetos, métodos e strings

**Variáveis de instância,  
métodos set e  
métodos get**



# Visualização conceitual de um objeto com dados encapsulados



# Inicialização de objetos com construtores

Ao declarar uma classe, você pode fornecer seu próprio construtor a fim de especificar a inicialização personalizada para objetos de sua classe. Por exemplo, você pode querer especificar um nome para um objeto “qualquer” quando ele é criado, como na linha abaixo:

```
Account account1 = new Account("Alexandre");
```

# Utilizando estrutura de repetição

Tem-se um conjunto de dados contendo a altura e o sexo “M” Masculino e “F” Feminino de 10 pessoas. Fazer um programa em JAVA que calcule e escreva:

- a) A maior e a menor altura do grupo;
- b) A média de altura dos homens;
- c) A quantidade de mulheres.

# O que são Arrays?

---

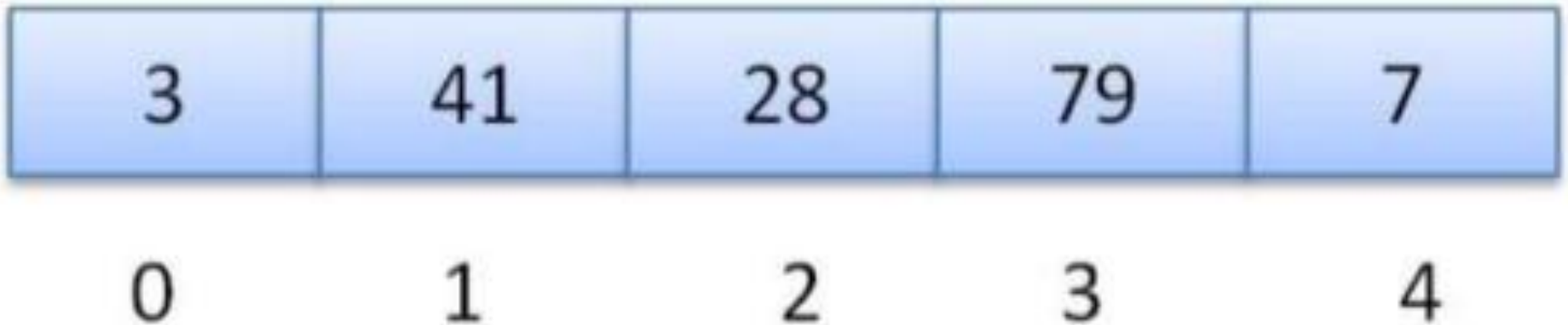


Figura 1: Um array de 5 elementos

# Declaração em JAVA

```
tipo[ ] nomeDoArray = new tipo[tamanhoDoArray];
```

```
Int[ ] numero = new int[10];
```

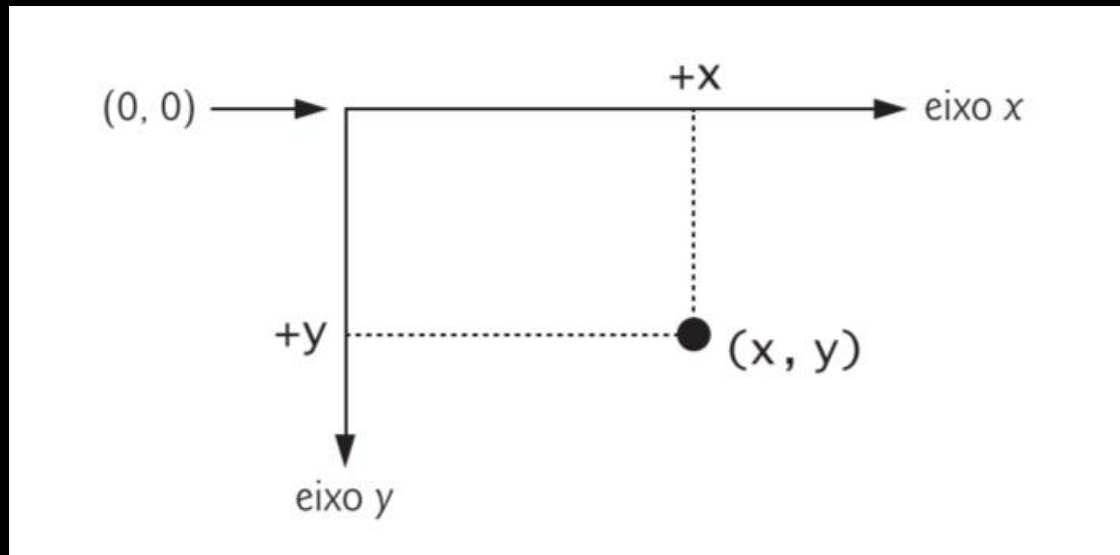
```
String[ ] nome = new String[5];
```

## Aplicando Array em um Exemplo

O usuário informa 5 números que são armazenados em um array. Em seguida, o programa imprime os números informados e calcula a média deles. Note que o programa usa a classe Scanner para ler os números informados pelo usuário.

# GUIs e imagens gráficas

## Criando desenhos simples



# GUIs e imagens gráficas: utilizando caixas de diálogo





## Exercício com GUI e imagens gráficas:

Faça um programa de adição, utilizando entrada e saída baseadas em caixas de diálogo com os métodos da classe `JOptionPane`. Uma vez que o método `showInputDialog` retorna uma `String`, você deve converter a `String` que o usuário insere em um `int` para utilização em cálculos. O método `static parseInt` da classe `Integer` (pacote `java.lang`) recebe um argumento `String` que representa um inteiro e retorna o valor como um `int`.

### Exemplo:

```
String num1 = .....;  
int convertNum1 = Integer.parseInt(num1);
```

# O que são Arrays?

Os arrays ou matrizes, como são conhecidos pelo Java, fazem parte do pacote `java.util`. São objetos de recipientes que contém um número fixo de valores de um único tipo. O comprimento de um array é estabelecido quando criado, sendo que após a criação o seu comprimento fica fixo.

Cada item em um array é chamado de elemento, e cada elemento é acessado pelo número, o índice.