

Parte 02

Prática do Hello, World!!

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

Exemplo Geral: Classe, Objeto, Atributos e Métodos

```
public class Casa {  
  
    int qtdQuartos;  
    String corCasa;  
    Double area;  
    int qtdPortas;  
    String cidade;  
  
    public void imprimeValores() {  
        System.out.print(qtdQuartos+"\n");  
        System.out.print(qtdPortas+"\n");  
        System.out.print(area+"\n");  
        System.out.print(corCasa+"\n");  
        System.out.print(cidade+"\n");  
    }  
}
```

```
public class Cidade {  
  
    public static void main(String[] args) {  
        Casa casa1 = new Casa();  
        casa1.qtdQuartos = 3;  
        casa1.qtdPortas = 10;  
        casa1.area = 100.7;  
        casa1.corCasa = "Branca";  
        casa1.cidade = "Fortaleza";  
        casa1.imprimeValores();  
    }  
}
```

Atividade para ser entregue:

Seguindo o exemplo anterior, faça:

- Crie a classe chamada **Indicadores** e um método chamado **ImprimirIndicadores** para guardar 3 valores inteiros que imprima na tela:

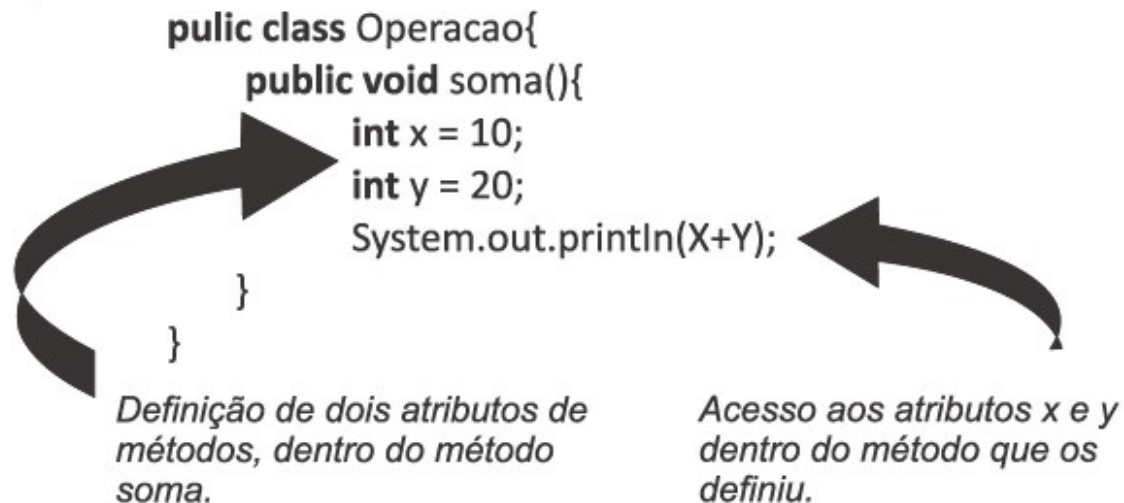
```
int num1, num2, num3;      public void imprimeIndicadores() {
```

- Em seguida crie uma classe chamada **ExecutarValores**: ela deve instanciar a classe **Indicadores** para um atributo chamado **indicador1** nele você deve colocar os três valores, e use o método **imprimirIndicadores** do método **Indicadores** para imprimir os 3 indicadores na tela.

Atributos de Método

Atributos definidos **dentro** de métodos, são chamados **atributos de método**. Estes podem ser acessados apenas localmente no método que os definiu.

Exemplo



Atributos de Instância

Atributos definidos **fora** de métodos são chamados de **atributos de instância**.

```
public class Funcionario{  
    String nome;  
    int idade;  
    double peso;  
    float altura;  
  
    public void alteraldade(){  
        idade = 25;  
    }  
}
```

*Acesso correto do atributo
idade dentro do método
alteraldade.*

```
public class Execucao{  
    public static void main(String args[]){  
        Funcionario f1 = new Funcionario();  
  
        f1.nome = "João"  
        f1.idade = 44;  
        f1.peso = 65.8;  
        f1.altura = 1.75f;  
    }  
}
```


*Acesso correto dos atributos
de Funcionario através de uma
instância.*

Atributos de Classe

Similar aos atributos de instância, a diferença é o uso da palavra reservada **static** antes do tipo de dados do atributo. São **compartilhados** por todas as instâncias, ou seja, a alteração no valor do atributo através de uma das instâncias reflete nas outras instâncias.

```
public class Funcionario{  
    static boolean plantaoColetivo = false;  
}
```

```
public class Execucao{  
    public static void main(String args[]){  
        Funcionario f1 = new Funcionario();  
        Funcionario f2 = new Funcionario();  
  
        f1.plantaoColetivo = true;  
        f2.plantaoColetivo = false;  
  
        System.out.println(f1.plantaoColetivo);  
    }  
}
```



A saída da tela exibirá false. mesmo tendo atribuído true utilizando f1. Lembre-se que o atributo de classe plantaoColetivo é compartilhado por todas as instâncias.

Métodos de Instância

A chamada do método de instância pode ser realizado dentro da própria classe ou através de uma instância da classe que o definiu.

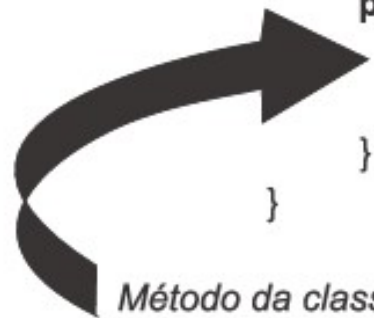
```
public class metodoInstancia{  
    public int soma(int x, int y){  
        return (x+y);  
    }  
  
    public void chamador(){  
        int result = soma(1,10);  
    }  
}
```

```
public class Exec{  
    public static void main(String args[]){  
        MetodoInstancia mi = new metodoInstancia();  
        int result = mi.soma(2,8);  
    }  
}
```


Métodos de Classe

Um método de classe pode acessar atributos de classe ou atributos definidos dentro dele mesmo, e um método de classe pode acessar apenas outros métodos de classe.

```
public class Execucao{  
    public static int soma(int x, int y){  
        return(x+y);  
    }  
  
    public static void main(String args[]){  
        int resultado = soma(9, 7);  
        System.out.println(resultado);  
    }  
}
```



Método da classe main acessando outro método de classe chamado soma.

Leitura de Dados do Teclado

- O comando de entrada é utilizado para receber dados digitados pelo usuário.
- Os dados recebidos devem ser armazenados em variáveis.
- Uma das formas de leitura de dados que Java disponibiliza é por meio da classe **Scanner**.
- É necessário a importar o pacote **Java.util** e instanciar de seus **objetos**

```
import java.util.*;
```

```
class LerTexto {
```

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

```
        Scanner entrada = new Scanner(System.in);
```

```
        String texto1;
```

```
        System.out.println("Escreva o texto: ");
```

```
        texto1 = entrada.next();
```

```
        System.out.println("O texto escrito foi: "+texto1);
```

```
    }
```

```
}
```

Leitura de Dados do Teclado

Dependendo do tipo de dados a ser lido, existe um método diferente na classe **Scanner**

Função	Funcionalidade
<code>next()</code>	Aguarda uma entrada em formato String com uma única palavra;
<code>nextLine()</code>	Aguarda uma entrada em formato String com uma ou várias palavras;
<code>nextInt()</code>	Aguarda uma entrada de formato inteiro;
<code>nextDouble</code>	Aguarda uma entrada em formato fracionário;

Leitura de Dados do Teclado

Soma de dois números inteiros passados pelo usuário

```
import java.util.*;

class Soma2 {

    public static void main(String[] args) {
        Scanner entrada = new Scanner(System.in);
        int a, b, soma;
        System.out.println("Escreva número1");
        a = entrada.nextInt();
        System.out.println("Escreva número2");
        b = entrada.nextInt();
        soma = a + b;
        System.out.println(soma);
    }
}
```

Operadores Lógicos, Aritméticos e Relacionais

Operador	Finalidade	Tipo de dados que trabalha	Valor retorno da operação
	ou	Booleano	Booleano
&	e	Booleano	Booleano
!	not	Booleano	Booleano
=	Atribuição	Todos	Depende dos Operandos
-, *, /	Operadores matemáticos	Numéricos	Depende dos Operandos
+	Soma números e concatena Strings	Numéricos ou String	Depende dos Operandos
%	Resto da divisão	Numérico	Numérico
==	Igualdade	Todos	Booleano
!=	Diferença	Todos	Booleano
>, <, >=, <=	Maior, Menor, Maior ou Igual, Menor ou Igual	Numéricos e Caractere	Booleano