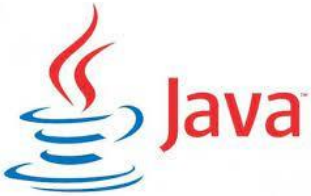
iwtraining



Formação Java - Aula 04

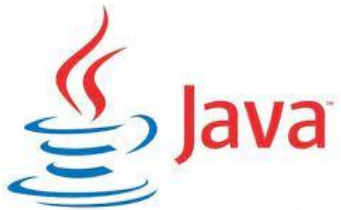
Professor: Leonardo Gomes.
WhatsApp/Telegram: (85) 98421-8509
Email: leonardobrendoti@gmail.com
IwTraining formações.

Sumário.

1. Variáveis e programação procedural;
2. interação via console, operadores, estruturas condicionais, operador ternário e Métodos;
3. Método de classe e de objeto; Listas e arrays; Estruturas de repetição; Recursividade.
4. Programação orientada a objetos;

iwtraining

Métodos e funções.

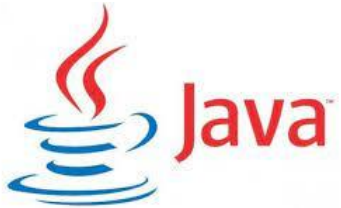


Principal.java

```
1
2  public class Principal {
3
4      public static void main(String[] args) {
5          // TODO Auto-generated method stub
6          matematica(10, 20);
7      }
8
9      public static void matematica(int num1, int num2) {
10         int resultado = num1 + num2;
11         System.out.println(resultado);
12     }
13
14 }
15
```

iwtraining

Interação via console.

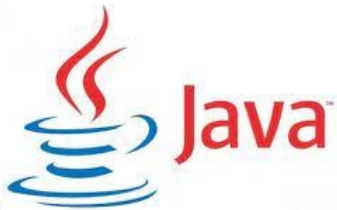


Main.java | Principal.java

```
1  import java.util.Scanner;
2  public class Principal {
3      public static void main(String[] args) {
4          Scanner entrada = new Scanner(System.in);
5          double numero1 = 0;
6          double numero2 = 0;
7          double resultado = 0;
8
9          System.out.println("Digite o número 1");
10         numero1 = entrada.nextDouble();
11         System.out.println("Digite o número 2");
12         numero2 = entrada.nextDouble();
13         resultado = numero1 + numero2;
14         System.out.println("A soma dos números digitados é: " + resultado);
15     }
16 }
```

iwtraining

Estrutura condicional - if e else exemplo.

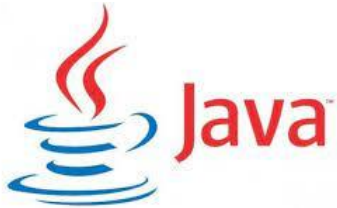


Main.java

```
1
2  public class Main {
3
4      public static void main(String[] args) {
5
6          int nota1 = 7;
7
8          if(nota1 > 7) {
9              System.out.println("Maior que 7");
10         }else if(nota1 < 7) {
11             System.out.println("Menor que 7");
12         }else {
13             System.out.println("Igual a 7");
14         }
15     }
16 }
17
```

iwtraining

Estrutura cond. - if,else anin. - exemplo.

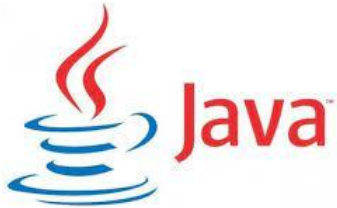


Main.java

```
1
2  public class Main {
3      public static void main(String[] args) {
4          double nota1 = 7.1;
5          if(nota1 > 7) {
6              if (nota1 <= 8) {
7                  System.out.println("Até 8");
8              }else if(nota1 <= 9) {
9                  System.out.println("Até 9");
10             }else {
11                 System.out.println("até 10");
12             }
13         }else if(nota1 < 7) {
14             System.out.println("Menor que 7");
15         }else {
16             System.out.println("Igual a 7");
17         }
18     }
19 }
```

iwtraining

Estrutura condicional - Switch - Exemplo.

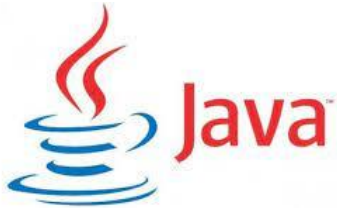


Main.java | Principal.java

```
1
2  public class Principal {
3      public static void main(String[] args) {
4          int nota1 = 1;
5          switch (nota1) {
6          case 1:
7              System.out.println("Opção 1");
8              break;
9          case 2:
10             System.out.println("Opção 2");
11             break;
12         case 3:
13             System.out.println("Opção 3");
14             break;
15         default:
16             System.out.println("Outras opções");
17             break;
18         }
19     }
20 }
```

iwtraining

Estrut. cond. - Switch aninhado - Exemplo.

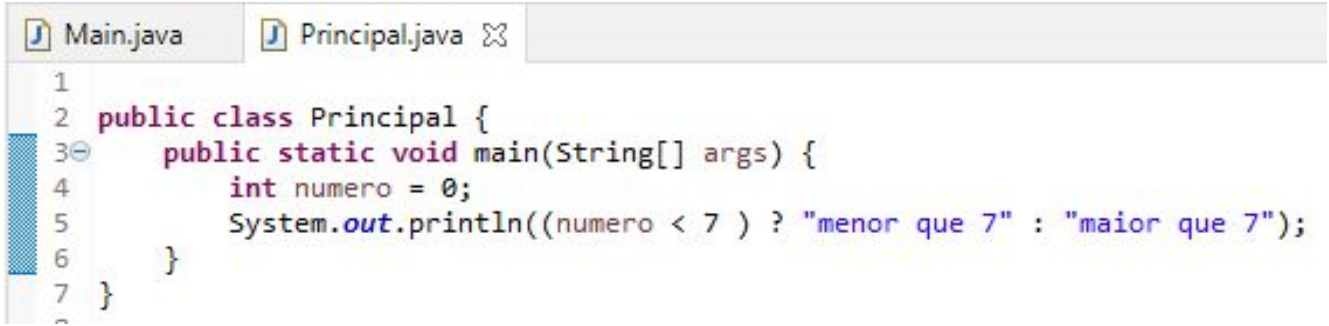


Main.java | Principal.java

```
1
2  public class Principal {
3      public static void main(String[] args) {
4          int nota1 = 1;
5          int indice = 8;
6          switch (nota1) {
7          case 1:
8              System.out.println("Opção 1");
9              if(indice < 7) {
10                 System.out.println("Indice menor que 7");
11             }else {
12                 System.out.println("Indice maior que 7");
13             }
14             break;
15         case 2:
16             System.out.println("Opção 2");
17             break;
18         case 3:
19             System.out.println("Opção 3");
20             break;
21         default:
22             System.out.println("Outras opções");
23             break;
24         }
25     }
26 }
```

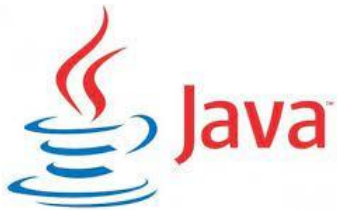
iwtraining

Operador ternário - exemplo.



iwtraining

Sobrecarga de métodos - Exemplo.

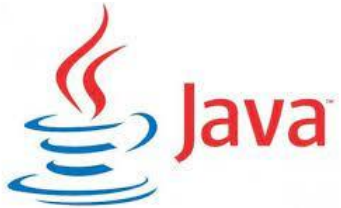


Main.java | Principal.java

```
13      }
14  }
15  class ClasseTeste3{
16      public static double subtrair(double num1, double num2) {
17          double resultado = 0;
18          resultado = num1 - num2;
19          return resultado;
20      }
21      public static int subtrair(int num1, int num2) {
22          int resultado = 0;
23          resultado = num1 - num2;
24          return resultado;
25      }
26
27      public static int subtrair(int num1, int num2, int num3) {
28          int resultado = 0;
29          resultado = num1 - num2 - num3;
30          return resultado;
31      }
32
33  }
34
35  public class Principal {
36      public static void main(String[] args) {
37          int numero = 0;
38          System.out.println("Valor da soma: " + ClasseTeste1.somar(2, 3));
39          ClasseTeste2 classeTeste2 = new ClasseTeste2();
40          System.out.println("Valor da subtração: " + classeTeste2.subtrair(2, 3));
41          System.out.println("Valor da subtração 1 da classe 3: " + ClasseTeste3.subtrair(20.3, 3.2));
42          System.out.println("Valor da subtração 2 da classe 3: " + ClasseTeste3.subtrair(10, 3));
43          System.out.println("Valor da subtração 3 da classe 3: " + ClasseTeste3.subtrair(10, 4, 4));
44      }
```

iwtraining

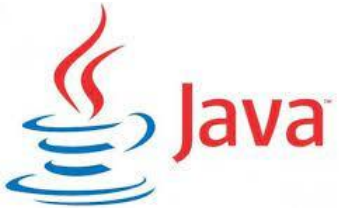
Método de classe - Exemplo




```java
class OperacaoMatematica{
    public static double adicao(double num1, double num2) {
        return num1 + num2;
    }
    public static double subtracao(double num1, double num2) {
        return num1 - num2;
    }
    public static double multiplicacao(double num1, double num2) {
        return num1 * num2;
    }
    public static double divisao(double num1, double num2) {
        return num1 / num2;
    }
}
public class Principal1 {
    public static void main(String[] args) {
        System.out.println("O valor da adição é: " + OperacaoMatematica.adicao(10,200));
        System.out.println("O valor da subtração é: " + OperacaoMatematica.subtracao(100,50));
        System.out.println("O valor da adição é: " + OperacaoMatematica.multiplicacao(10,40));
        System.out.println("O valor da adição é: " + OperacaoMatematica.divisao(150,30));
    }
}
```

iwtraining

Método de objeto - exemplo.

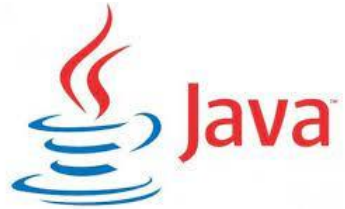



```java
class OperacaoMatematica{
    public  double adicao(double num1, double num2) {
        return num1 + num2;
    }
    public double subtracao(double num1, double num2) {
        return num1 - num2;
    }
    public double multiplicacao(double num1, double num2) {
        return num1 * num2;
    }
    public double divisao(double num1, double num2) {
        return num1 / num2;
    }
}
public class Principal1 {
    public static void main(String[] args) {
        OperacaoMatematica operacaoMatematica = new OperacaoMatematica();

        System.out.println("O valor da adição é: " + operacaoMatematica.adicao(10,200));
        System.out.println("O valor da subtração é: " + operacaoMatematica.subtracao(100,50));
        System.out.println("O valor da adição é: " + operacaoMatematica.multiplicacao(10,40));
        System.out.println("O valor da adição é: " + operacaoMatematica.divisao(150,30));
    }
}
```

iwtraining

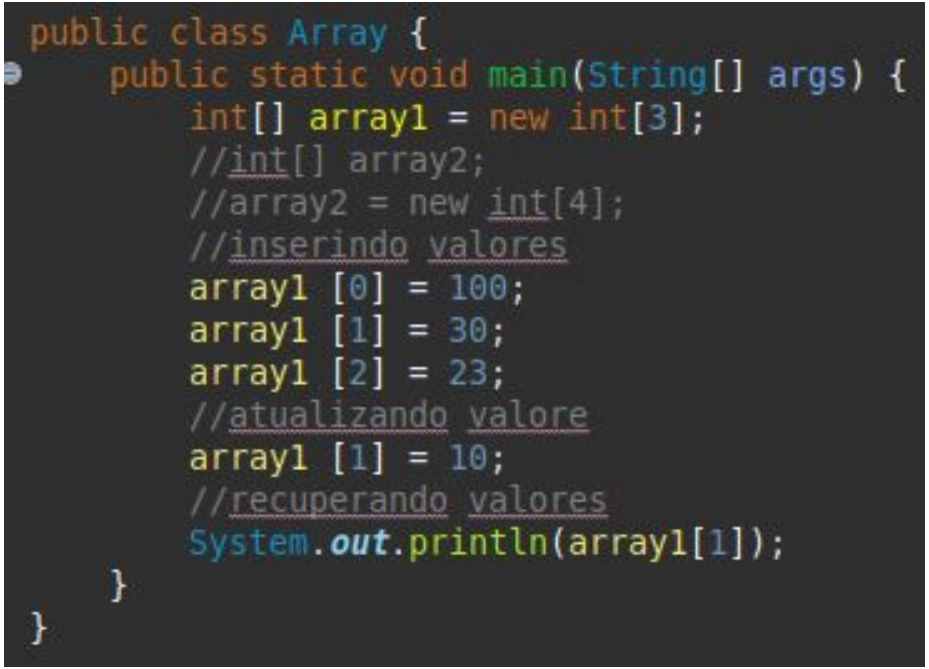
Listas - Exemplo




```java
import java.util.ArrayList;
public class Lista {
    public static void main(String[] args) {
        ArrayList<String> endereco = new ArrayList<>();
        //adicionando valores
        endereco.add("Rua do Bobo");
        endereco.add("Avenida 13 de maio");
        endereco.add("Avenida Pontes Vieira");
        //removendo valores
        endereco.remove(2);
        //recuperando valores
        endereco.get(0);
        //atualizando valores
        endereco.set(1, "Rua da quadra");
        System.out.println(endereco);
    }
}
```

iwtraining

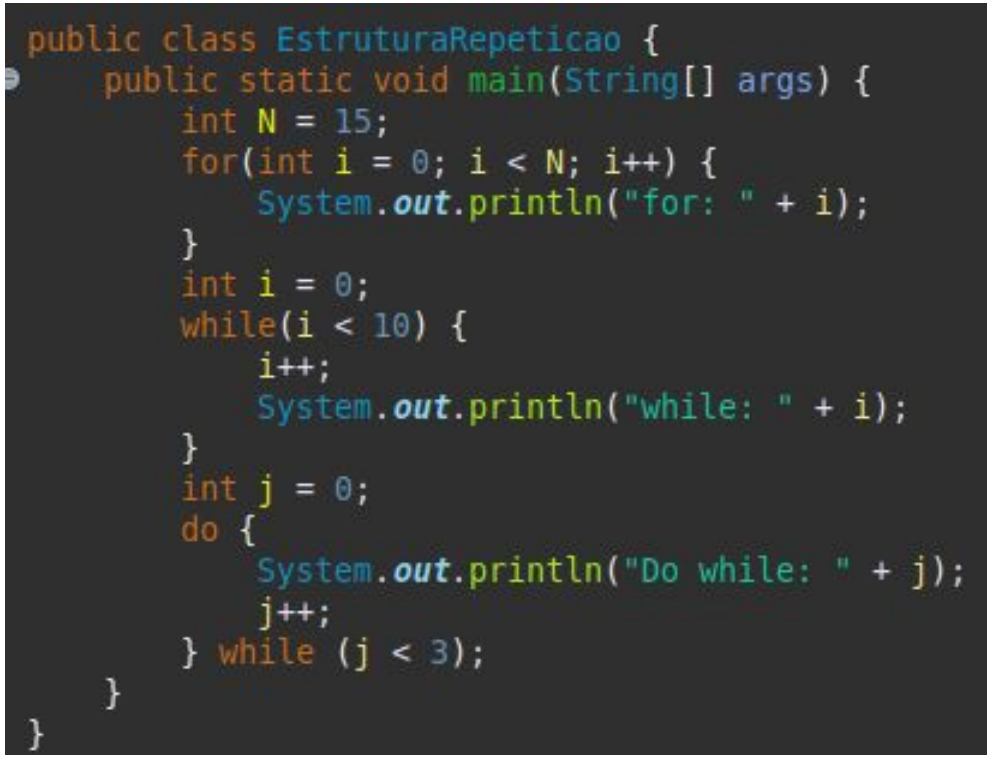
Array - Exemplo



```java
public class Array {
    public static void main(String[] args) {
        int[] array1 = new int[3];
        //int[] array2;
        //array2 = new int[4];
        //inserindo valores
        array1 [0] = 100;
        array1 [1] = 30;
        array1 [2] = 23;
        //atualizando valore
        array1 [1] = 10;
        //recuperando valores
        System.out.println(array1[1]);
    }
}
```

iwtraining

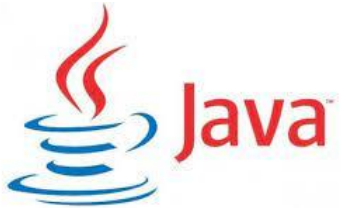
Estrutura. de repet. - for, while e do while



```java
public class EstruturaRepeticao {
    public static void main(String[] args) {
        int N = 15;
        for(int i = 0; i < N; i++) {
            System.out.println("for: " + i);
        }
        int i = 0;
        while(i < 10) {
            i++;
            System.out.println("while: " + i);
        }
        int j = 0;
        do {
            System.out.println("Do while: " + j);
            j++;
        } while (j < 3);
    }
}
```

iwtraining

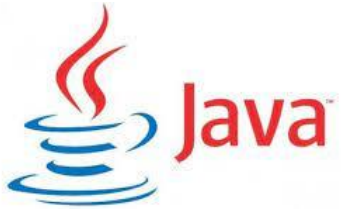
Recursividade - exemplo




```java
class OperacaoRecursiva{
    public static int somarPares(int x) {
        if(x == 0) return 0;
        if(x % 2 == 0) return x + somarPares(x - 1);
        return somarPares(x - 1);
      }
}
public class Recursividade {
    public static void main(String[] args) {
        System.out.println(OperacaoRecursiva.somarPares(5));
    }
}
```

iwtraining

Programação orientada a objetos



Programação orientada a objetos (POO, ou OOP segundo as suas siglas em inglês) é um paradigma de programação baseado no conceito de "objetos", que podem conter dados na forma de campos, também conhecidos como atributos, e códigos, na forma de procedimentos, também conhecidos como métodos.

Uma característica de objetos é que um procedimento de objeto pode acessar, e geralmente modificar, os campos de dados do objeto com o qual eles estão associados (objetos possuem uma noção de "this" (este) ou "self" (próprio)).

iwtraining

Programação procedural e programação orientada a objetos

iwtraining

Conceitos importantes em POO

- Classe;
- Get e set;
- construtor;
- Objeto;
- Métodos e sobrecarga;
- Abstração;
- Encapsulamento;
- Modificadores de acesso;
- Herança;
- Polimorfismo;
- Associação de classes;
- Composição/Agregação;
- especialização/generalização;
- Interface x classe abstrata;
- UML.

iwtraining