

INSTITUTO FEDERAL DE  
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
SÃO PAULO  
Campus São Carlos

*Análise e Desenvolvimento de Sistemas*

*Programação Orientada a Objetos (POOS3)*

# Introdução à Java



**Pablo Dalbem**

**[dalbem@ifsp.edu.br](mailto:dalbem@ifsp.edu.br)**

# Objetivos

Ter uma noção de como funciona a plataforma Java.

Desenvolver programas usando a linguagem Java no IDE IntelliJ (<https://www.jetbrains.com/pt-br/idea/>)

# Conteúdo

Plataforma Java

Tipos de dados

Entrada e saída de dados

Estruturas de decisão

Estruturas de repetição

Arranjos

Exercícios

# Java

## Oracle Java

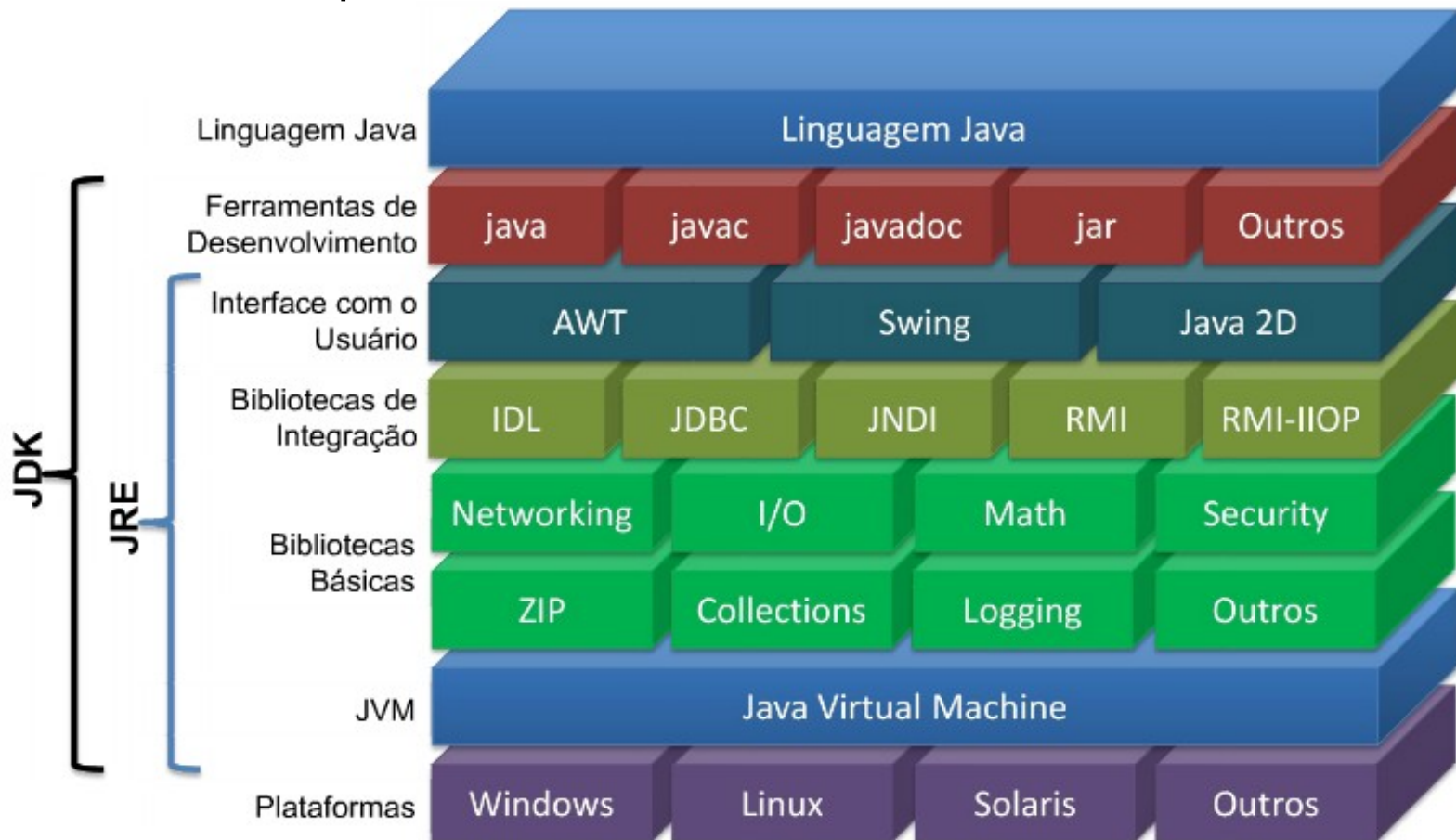
---

Java is the #1 programming language and development platform. It reduces costs, shortens development timeframes, drives innovation, and improves application services. With millions of developers running more than 51 billion Java Virtual Machines worldwide, Java continues to be the development platform of choice for enterprises and developers.

# Plataforma Java

**JRE** – Java Runtime Environment

**JDK** – Java Development Kit



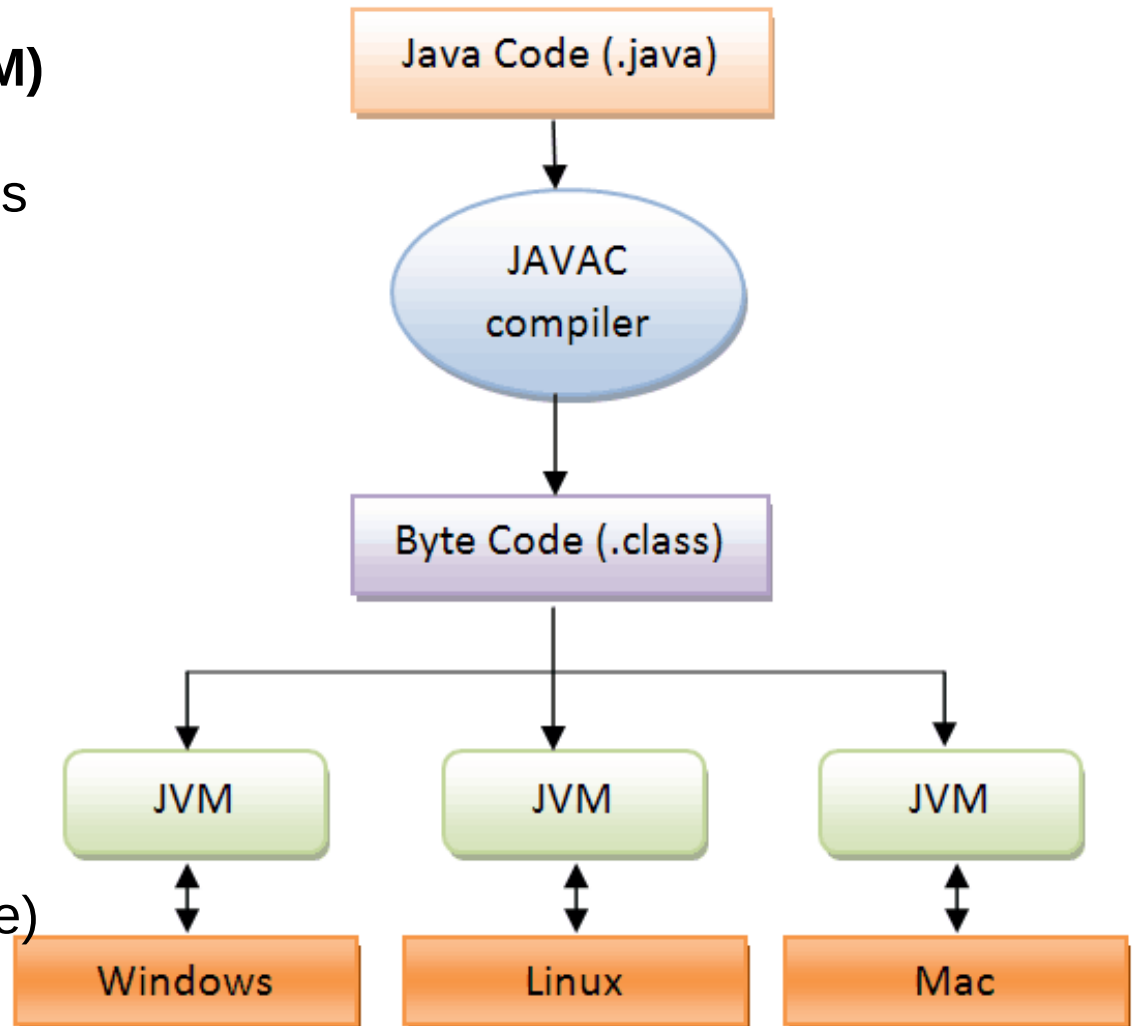
# Plataforma Java

## Java Virtual Machine (JVM)

Camada intermediária responsável por executar os programas em Java.

## Princípio WORA

(Write Once, Run Anywhere)



# Programa em Java

```
package primeiroprograma;
```

```
public class Main {
```


```
    public static void main(String[] args) {  
        // write your code here  
    }
```

```
}
```

# Programa em Java

```
package primeiroprograma;
```

Namespace  
para organizar  
classes  
relacionadas



```
public class Main {
```

```
    public static void main(String[] args) {  
        // write your code here  
    }
```

```
}
```

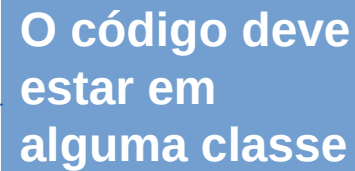


# Programa em Java

```
package primeiroprograma;
```

```
public class Main {
```

O código deve  
estar em  
alguma classe



```
    public static void main(String[] args) {  
        // write your code here  
    }
```

```
}
```

# Programa em Java

```
package primeiroprograma;
```

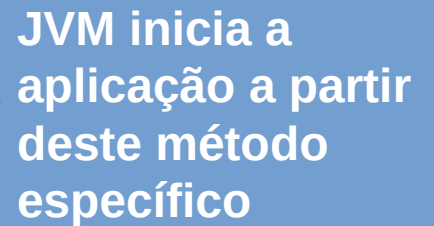
```
public class Main {
```

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

```
        // write your code here
```

```
    }
```

```
}
```



JVM inicia a aplicação a partir deste método específico

# Tipos Primitivos de Dados

Tipo	Descrição	Tamanho e Intervalo	Valor padrão
boolean	Valor lógico	---	false
char	Caractere unicode	0 e 65535 ( $2^{16}$ )	'\u0000'
byte	Inteiro de 8 bits	$-2^7 = -128$ a $2^7-1 = 127$	0
short	Inteiro de 16 bits	$-2^{15} = -32.768$ a $2^{15}-1 = 32.767$	0
int	Inteiro de 32 bits	$-2^{31} = -2.147.483.648$ a $2^{31}-1 = 2.147.483.647$	0
long	Inteiro de 64 bits	$-2^{31}$ a $2^{31}$	0L
float	Ponto flutuante de precisão simples	$1.40239846 \cdot 10^{-46}$ a $3.40282347 \cdot 10^{38}$	0.0f
double	Ponto flutuante de precisão dupla	$4.94065645841246544 \cdot 10^{-324}$ a $1.7976931348623157 \cdot 10^{308}$	0.0d

# Tipos de Dados

```
public class Main {  
    public static void main(String[] args) {  
        boolean varBoolean;  
        varBoolean = true;  
  
        byte varByte = 127;  
  
        short varShort = 32767;  
        int varInt = 2147483647;  
        long varLong = 9223372036854775807L;  
  
        float varFloat = 3.141592f;  
        double varDouble = 321321.3123435;  
    }  
}
```

# Constantes

Ao contrário das variáveis, as constantes precisam ser inicializadas e não podem ter o seu valor alterado.

Para declarar um atributo como constante, utilize o modificador **final**.

Por convenção, constantes são definidas em CAIXA ALTA,

**Exemplos:**

```
final int QTD_DIAS = 10;
```

```
final boolean CLIENTE_ATIVO = true;
```

```
final double VALOR;
```

```
VALOR = 15.5;
```

# Enumeração

**Enumeração (enum):** permite limitar o intervalo de valores aceitos por uma variável por meio de um conjunto de constantes.

Enum devem ser utilizado sempre que for necessário representar conjuntos finitos, como dias da semana, meses, estados de pedidos, etc.

A variável do tipo enum deve ser igual a um dos valores predefinidos para ela.

## Exemplo 1:

```
enum FormaPagamento {DINHEIRO, DEBITO, CREDITO}
```

```
FormaPagamento pagamento = FormaPagamento.DINHEIRO;
```

```
...
```

```
pagamento = FormaPagamento.CREDITO;
```

# Enumeração

**Exemplo 2:**

```
public class Main {  
  
    public static void main(String[] args) {  
        enum DiaSemana {SEGUNDA, TERÇA, QUARTA, QUINTA, SEXTA}  
  
        DiaSemana hoje = DiaSemana.QUINTA;  
    }  
}
```

# Operadores

Categoria	Operadores
Unário	expr++   expr--   ++expr   --expr   +expr   -expr   !
Binário aditivo	+   -
Binário multiplicativo	*   /   %
Binário relacional	==   !=   <   >   <=   >=   instanceof
Binário lógico	&&
Lógico bit a bit (bitwise)	&   ^
Deslocamento	<<   >>   >>>
Ternário	? :
Atribuição	=   +=   -=   *=   /=   %=   &=   ^=  =   <<=   >>=   >>>=



# Entrada e Saída de Dados

```
import java.util.Scanner;
```

```
public class Main {  
    public static void main(String[] args) {  
        Scanner entrada = new Scanner(System.in);  
  
        System.out.print("Digite o nome: ");  
        String nome = entrada.nextLine();  
        System.out.println("Olá, " + nome);  
  
        int idade = entrada.nextInt();  
        float salario = entrada.nextFloat();  
  
    }  
}
```

# Entrada e Saída de Dados

É possível formatar a saída usando System.out.printf

Exemplo de formatação para um número float com 3 casas decimais:

```
float valor = 10.123456f;
```

```
System.out.printf("Saída formatada: %.3f",valor); //imprime 10,123
```

Material sobre formatação:

<https://docs.oracle.com/javase/tutorial/java/data/numberformat.html>

# Estruturas de Decisão

## IF

```
if (<expressao_logica>){  
    <comandos>;  
} else {  
    <comandos>;  
}
```

# Estruturas de Decisão

## Switch-Case

```
switch (<seletor>){  
    case <value_1>:  
        <comandos>  
        break;  
    case <value_n>:  
        <comandos>  
        break;  
    default:  
        <comandos>  
        break;  
}
```

# Switch Case - Exemplo

```
public class ExemploSwitchCase {  
    public static void main(String[] args) {  
        int dia = 3;  
        switch (dia) {  
            case 1:  
                System.out.println("Domingo");  
                break;  
            case 2:  
                System.out.println("Segunda-feira");  
                break;  
            case 3:  
                System.out.println("Terça-feira");  
                break;  
            case 4:  
                System.out.println("Quarta-feira");  
                break;
```

```
            case 5:  
                System.out.println("Quinta-feira");  
                break;  
            case 6:  
                System.out.println("Sexta-feira");  
                break;  
            case 7:  
                System.out.println("Sábado");  
                break;  
        }  
    }  
}
```

# Estruturas de Repetição

```
while(<expressao_condicional>) {  
    <comandos>;  
}
```

```
do {  
    <comandos>;  
} while(<expressao_condicional>);
```

```
for(<inicializador>;<condicao_de_parada>;<iterador>) {  
    <comandos>;  
}
```

# Arranjos (Arrays)

Estruturas de dados que armazenam um número fixo de variáveis do mesmo tipo.

Cada elemento do array pode ser acessado por um índice.

Também chamados de **variáveis indexadas**, **vetores** (caso unidimensional) ou **matrizes** (caso multidimensional)

# Arranjo Unidimensional

**Exemplo de arranjo unidimensional (vetor):**

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        int[] vetor = new int[10];
        Scanner entrada = new Scanner(System.in);
        for (int i=0; i<10;i++)
            vetor[i] = entrada.nextInt();

        for (int i=0; i<10;i++)
            System.out.println(vetor[i]);

    }
}
```



# Arranjo Bidimensional

**Exemplo arranjo bidimensional (matriz):**

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

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

        Scanner entrada = new Scanner(System.in);

        for (int i=0; i<5;i++)
            for (int j=0; j<5;j++)
                matriz[i][j] = entrada.nextInt();

    }
}
```

# Arranjos (Arrays)

```
public class Main {  
    public static void main(String[] args) {  
  
        String[] linguagens = {"Java", "C", "Python"};  
  
        // for aprimorado  
        for (String ling : linguagens)  
            System.out.println(ling);  
    }  
}
```

# Exercícios

1. Faça um programa em Java que calcule o montante acumulado em um investimento. O usuário deverá fornecer como entradas: capital aplicado, taxa de juros anual (formato decimal) e o tempo da aplicação (em anos).

Fórmula dos juros compostos:

$$M = C * (1 + taxa) ^ tempo$$

Exemplo: Montante acumulado para uma aplicação inicial de 10.000 a uma taxa de 15% aa durante 10 anos

$$M = 10000 * (1 + 0.15) ^ 10$$

$$M = 10000 * (1.15) ^ 10$$

$$M = 10000 * 4.0455577$$

$$M = 40455,58$$

**DICA:** pesquise por `Math.pow()` para efetuar potenciação

# Exercícios

2. Escreva um programa em Java que receba 3 números inteiros representando as medidas dos lados de um triângulo.

O programa deverá verificar se os números lidos formam um triângulo.

Dado um triângulo com lados **a**, **b** e **c**, esse triângulo somente existirá se:

$$a + b > c$$

$$a + c > b$$

$$b + c > a$$

Se os números lidos formarem um triângulo, o programa deverá mostrar qual seu tipo:

Equilátero (3 lados iguais)

Isósceles (2 lados iguais)

Escaleno (3 lados diferentes)

# Exercícios

**3.** Faça um programa em Java que leia 5 notas de um aluno. Em seguida, o programa deverá calcular a média destas notas e mostrar quais delas estão acima desta média.

DICA: Você deverá utilizar arranjo de inteiros.

# Exercícios

4. Faça um programa em Java que leia 2 arrays A e B, de cinco posições cada, para armazenar números inteiros. Crie um terceiro array C composto pela soma dos números contidos em A e B.

A

<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
----------	----------	----------	----------	----------

B

<b>10</b>	<b>20</b>	<b>30</b>	<b>40</b>	<b>50</b>
-----------	-----------	-----------	-----------	-----------

C

<b>11</b>	<b>22</b>	<b>33</b>	<b>44</b>	<b>55</b>
-----------	-----------	-----------	-----------	-----------