



*Linguagem de Programação*  
***JAVA – Sintaxe Básica***

Profa. Joyce Miranda

# Variáveis Primitivas

---



## ▶ Tópicos

- ▶ declaração, atribuição de valores, casting e comparação de variáveis;
- ▶ controle de fluxo através de if e else;
- ▶ instruções de laço for e while, controle de fluxo com break e continue.

# Estrutura do código JAVA

---

## **Primeiro Programa em Java**

Arquivo: PrimeiroPrograma.java

```
public class PrimeiroPrograma {  
    public static void main( String[] args ) {  
        System.out.println( "Meu primeiro programa em Java" );  
    }  
}
```

**Compilando o código-fonte:**

```
javac PrimeiroPrograma.java
```

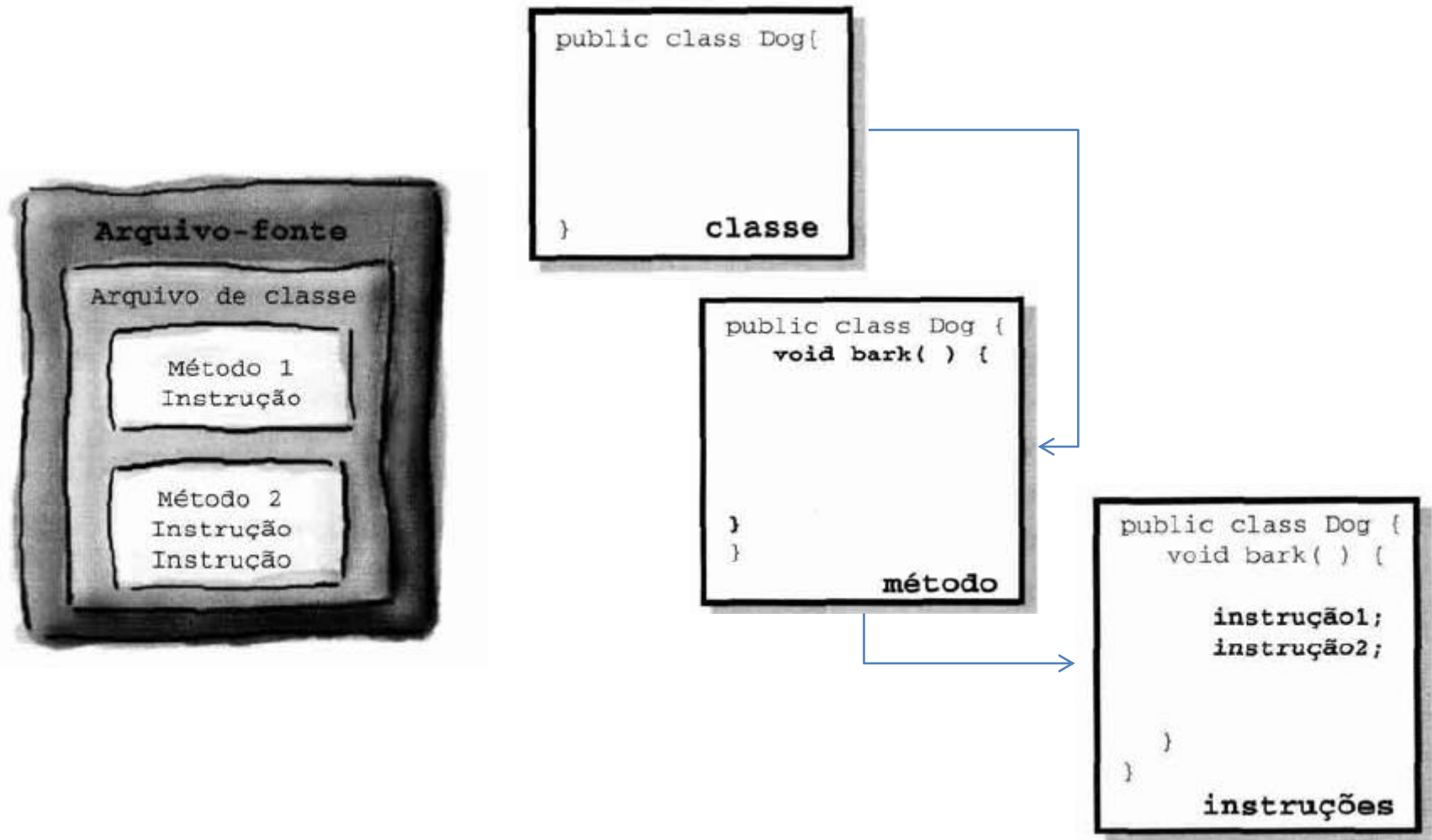
**Executando o programa:**

```
java PrimeiroPrograma
```

**Saída gerada:**

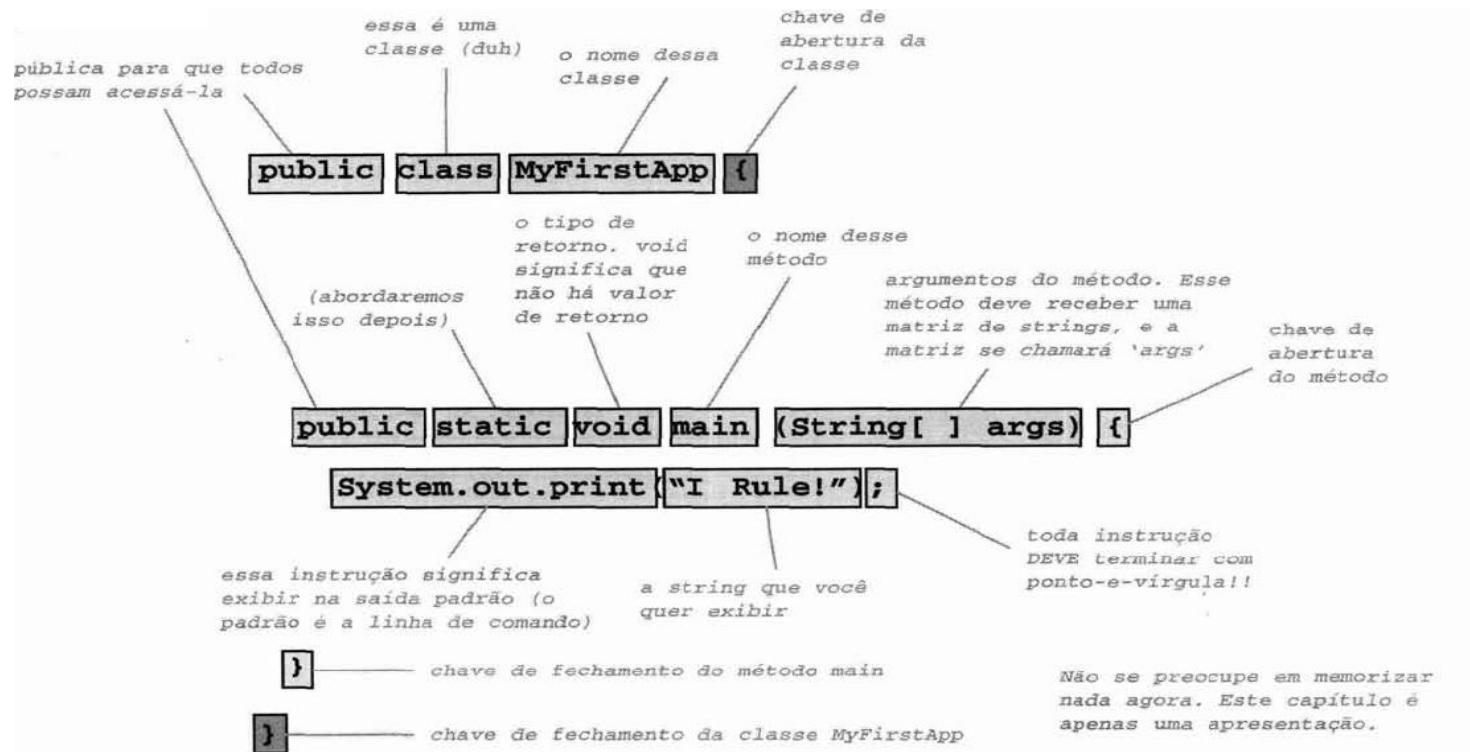
```
Meu primeiro programa em Java
```

# Estrutura do código JAVA



# Estrutura do Código

- ▶ Todo classe JAVA executável precisa ter:
  - ▶ A declaração de uma classe
  - ▶ Um método *main*



## Variáveis Primitivas

---

- ▶ Declaração de Variáveis

- ▶ **tipoDaVariavel** nomeDaVariavel;

- ▶ Ex: **int** idade;

- ▶ Atribuição de Valores

- ▶ **idade = 32;**

- ▶ Comentários

- ▶ **//** para comentar até o final da linha

- ▶ **/\* \*/** para comentar o que estiver entre eles

```
/* comentário daqui,  
ate aqui */  
  
//uma linha de comentário sobre a idade  
int idade;
```

## Tipos Primitivos e Valores

---

- ▶ Tipos utilizados e seu respectivo tamanho

<b>TIPO</b>	<b>TAMANHO</b>
boolean	1 bit
byte	1 byte
short	2 bytes
char	2 bytes
int	4 bytes
float	4 bytes
long	8 bytes
double	8 bytes

# Operadores

---

## ► Operadores Aritméticos

### Multiplicação e Divisão: \* e /

```
int um = 3 / 2;           // divisão de inteiros gera um inteiro
float umEmeio = (float) 3 / 2; // ocorre promoção aritmética para float
double xyz = umEmeio * um; // ocorre promoção aritmética para float
```

### Módulo: %

```
int resto = 7 % 2;           // resto = 1
```

### Adição e Subtração: + e -

```
long l = 1000 + 4000;
double d = 1.0 - 0.01;
```

### Concatenação:

```
long var = 12345;
String str = "O valor de var é " + var;
```



# Operadores

---

## ► Operadores de Comparação

### **Comparação ordinal: >, >=, < e <=**

Compara tipos primitivos numéricos e o tipo char.

```
boolean b = ( 10 < 3 );  
boolean w = (x <= y);  
if( x >= y ) { }
```

### **Comparação de Igualdade: == e !=**

Comparam tipos primitivos, valores literais e referências de objetos.

```
if( abc == 10 ) { }  
boolean b = ( xyz != 50 );  
if( refObj1 == refObj2 ) { }
```

# Operadores

---

## ▶ Operadores Lógicos

▶ (e) &&

▶ (ou) ||

▶ (não) !

▶ int A = 10;

▶ int B = 5;

▶ if ( A > 10 || B > 5 ) { Sout("ok!"); } else { Sout("não ok!"); } (?)

▶ boolean passou = true;

▶ if (!passou) {Sout("reprovado"); } else { Sout("aprovado!"); } (?)

## Casting e Promoção

---

### ► Conversão de Valores

- Para realizar atribuições com tipos incompatíveis sem que haja o erro de compilação, é preciso ordenar que o valor que será recebido seja moldado (*casted*) com o tipo da variável que irá recebê-lo.

### ► Ex.

```
double d3 = 3.14;  
int i = (int) d3;
```

# Operadores

---

## ▶ Trabalhando com Strings

### ▶ Declaração

- ▶ **String** nome = "Joyce Miranda" ;

### ▶ Comparação

- ▶ nome.**equals**("Joyce Miranda");

### ▶ Conversão

- ▶ String **valor** = "100";

- ▶ int A = **Integer.parseInt(valor)**;

- ▶ double B = **Double.parseDouble (valor)**;

- ▶ float C = **Float.parseFloat (valor)**;

## Estrutura Sequencial

---

### ► Praticando...

#### ► Considere que:

- Uma professora aplicou três avaliações referentes a uma disciplina.
- Para cada avaliação foi aplicado um peso específico.
- A 1ª avaliação teve peso 1, a 2ª avaliação teve peso 2 e a 3ª avaliação teve peso 3.
- Um aluno chamado **João Pedro** obteve as notas: **6.0**, **5.0** e **3.0** referentes respectivamente a 1ª, 2ª e 3ª avaliações.

- Crie um programa que imprima o nome do aluno e a média ponderada referente às notas obtidas na disciplina.

# Controles de Fluxo do Programa

## ► If-Else

```
if (condicaoBooleana) {  
    codigo;  
}
```

```
public class ClausulaIf {  
    public static void main( String[] args ) {  
        int idade = 20;  
        if( idade <= 12 ) {  
            System.out.println( "Criança" );  
        }  
        else if( idade <= 19 ) {  
            System.out.println( "Adolescente" );  
        }  
        else if( idade <= 60 ) {  
            System.out.println( "Adulto" );  
        }  
        else {  
            System.out.println( "Idoso" );  
        }  
    }  
}
```

# Controles de Fluxo do Programa

---

## ► While

```
public class LacoWhile {  
    public static void main( String[] args ) {  
        int i = 0;  
        //laço while() com bloco de código definido  
        while( i < 10 ) {  
            System.out.println( "Linha: " + i );  
            i++;  
        }  
    }  
}
```

# Controles de Fluxo do Programa

---

## ► Do-While

```
public class LacoWhile {  
    public static void main( String[] args ) {  
        int i = 0;  
        //laço do / while() com bloco de código definido  
        do {  
            System.out.println( "Linha: " + i );  
            i++;  
        } while( i < 10 );  
    }  
}
```



# Controles de Fluxo do Programa

---

## ► For

```
for (inicializacao; condicao; incremento) {  
    codigo;  
}
```

```
public class LacoFor {  
    public static void main( String[] args ) {  
        for( int i=0; i < 10; i++ ) {  
            System.out.println( "Linha: " + i );  
        }  
    }  
}
```

## Controles de Fluxo do Programa

---

### ► Break

- Aborta a execução de um laço, quando executado.

```
public class ClausulaBreak {  
    public static void main( String[] args ) {  
        char letras[] = { 'A', 'B', 'C', 'D', 'E' };  
        int i;  
        for( i=0; i<letras.length; i++ ) {  
            if( letras[i] == 'C' ) {  
                break;  
            }  
        }  
        System.out.println( "Último índice: " + i );  
    }  
}
```

## Controles de Fluxo do Programa

---

### ► Continue

- Ignora a execução dos comandos seguintes do bloco, no laço, quando executado.

```
public class ClausulaContinue {  
    public static void main( String[] args ) {  
        char letras[] = { 'B', 'X', 'R', 'A', 'S', 'I', 'L' };  
        int i;  
        for( i=0; i<letras.length; i++ ) {  
            if( letras[i] == 'X' ) {  
                continue;  
            }  
            System.out.print( letras[i] );  
        }  
    }  
}
```

## Controles de Fluxo do Programa

### ► Switch – Seleção Encadeada

```
public class ClausulaSwitch {  
    public static void main( String[] args ) {  
        int numero = 1;  
        switch( numero ) {  
            case 1 :  
                System.out.println( "UM" );  
                break;  
            case 2 :  
                System.out.println( "DOIS" );  
                break;  
            case 3 :  
                System.out.println( "TRES" );  
                break;  
            default :  
                System.out.println( "NENHUM" );  
                break;  
        }  
    }  
}
```

# Estrutura Condicional e Estrutura de Repetição

---

## ▶ Praticando...

### ▶ Implemente

- ▶ Defina valores para as variáveis X e Y; onde  $X < Y$ .
- ▶ Ao final imprima os números pares presentes no intervalo entre X e Y.

## Declaração de Vetores

- ▶ Na declaração de vetores deverão ser fornecidas três informações:
  - ▶ o nome do vetor;
  - ▶ o número de posições do vetor (seu tamanho);
  - ▶ o tipo de dado que será armazenado no vetor.

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

- ▶ **v** é declarado com um vetor de inteiros
- ▶ **new int[10]** aloca espaço na memória e cria efetivamente um vetor de inteiros, de tamanho 10.

*O Java é zero-based*

**v[10]**: posições de 0 a 9.

## Atribuição em Vetores

---

- ▶ Lembrando que:

$$0 \leq i \leq (\text{tam}-1)$$

```
int x[] = new int[5];  
//atribuição  
x[0] = 1;  
x[1] = 2;  
x[2] = x[1] * 2;  
x[3] = x[0];  
x[4] = x[2] + x[3];  
//escrita  
for(int i=0; i < x.length; i++){  
    System.out.println("x["+i+"] = " + x[i]);  
}
```

## Percorrendo Vetores

---

### ► *Enhanced for*

```
class AlgumaClasse {  
    void imprimeArray(int[] array) {  
        for (int x : array) {  
            System.out.println(x);  
        }  
    }  
}
```

```
int[] array = new int[10];  
for (int aux : array) {  
    System.out.println(aux);  
}
```



# Matrizes

---

- ▶ São variáveis indexadas com duas dimensões.
- ▶ A declaração de uma matriz na linguagem JAVA é a seguinte:

```
tipo [][] nomeDaMatriz
```

```
int [][] matrizDeInteiros;
```

- ▶ Para se criar efetivamente uma matriz deve-se utilizar o operador **new**.

```
nomeDaMatriz = new tipo[tamanho1][tamanho2];
```

```
matrizDeInteiros = new int[10][5];
```

# Matrizes

---

## ► Declaração em conjunto

```
tipo [][] nomeDaMatriz = new tipo[tamanho1][tamanho2];
```

```
int[][] matrizDeInteiros = new int[10][5];
```

## ► Percorrendo a matriz

```
for (int i = 0; i < 10; i++)  
    for (int j = 0; j < 5; j++)
```

## Vetores e Matrizes

---

### ► Praticando...

- Declare um vetor de 10 elementos;
- Preencha o vetor com números aleatórios entre 0 e 100;
- Imprima uma mensagem informando se o número **100** está presente ou não dentro do vetor.

## Entrada & Saída

---

### ► Passagem de Parâmetros

```
class PassagemParametro{  
    public static void main(String args[]){  
        System.out.println(args[0])        ;  
        System.out.println(args[1])        ;  
    }  
}
```

```
D:\Joyce\Material de Aula\Java & Web\Progs>java PassagemParametro 1 2  
1  
2
```

## Entrada & Saída

---

- ▶ Passagem de Parâmetros
  - ▶ Conversão de Classes

```
class PassagemParametro_Cast{  
    public static void main(String args[]){  
        int a = Integer.parseInt(args[0]);  
        int b = Integer.parseInt(args[1]);  
  
        int soma = a + b;  
  
        System.out.println("Soma: " + soma);  
    }  
}
```

## Entrada & Saída

---

### ▶ Entrada de Dados pelo Teclado

#### ▶ Scanner

**int** : `nextInt()`;  
**float**: `nextFloat()`;  
**double**: `nextDouble()`;  
**char**: `nextChar()`;  
**String**: `nextLine()`;

```
import java.util.Scanner;

public class Leitura_Scanner {

    public static void main(String[] args) {
        Scanner read = new Scanner(System.in);
        String nome; int idade;
        System.out.print("Escreva seu nome: ");
        nome = read.nextLine();
        System.out.print("Escreva sua idade: ");
        idade = read.nextInt();
        System.out.print(nome +
            " voce tem " + idade + " anos");
    }
}
```

## Entrada & Saída

```
import javax.swing.JOptionPane;

public class MediaAritmeticaJOptionPane {
    public static void main(String args[]){
        double n1, n2, n3, media;
        String entrada;

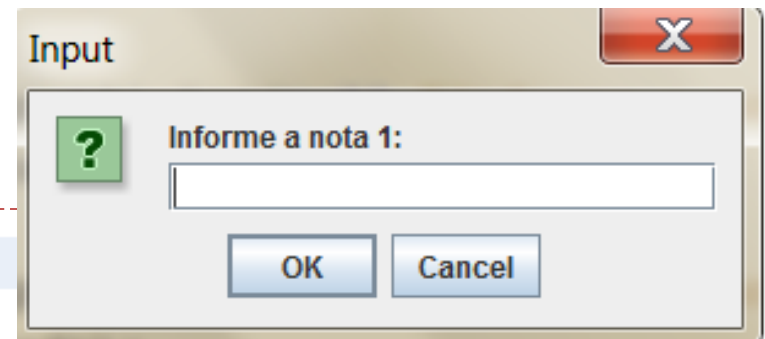
        entrada = JOptionPane.showInputDialog(null, "Informe a nota 1:");
        n1 = Double.parseDouble(entrada);

        entrada = JOptionPane.showInputDialog(null, "Informe a nota 2:");
        n2 = Double.parseDouble(entrada);

        entrada = JOptionPane.showInputDialog(null, "Informe a nota 3:");
        n3 = Double.parseDouble(entrada);

        media = (n1 + n2 + n3)/3;

        JOptionPane.showMessageDialog(null, "A média das notas é: " + media);
    }
}
```



## Entrada & Saída

---

### ► Praticando...

1. (*Scanner*) Leia um conjunto de nomes enquanto não for digitada a palavra “FIM”. Ao final informe a quantidade de vezes que o nome “MARIA” foi digitado.
2. (*JOptionPane*) A professora passou três avaliações e disse que iria eliminar a menor nota para calcular a média final. Com base nessa regra, faça um programa capaz de ler as três notas e calcular a média final referente a um aluno.



## Entrada & Saída

---

- ▶ Tarefa de implementação I (nota parcial).
  1. Escolha e resolva três questões de cada Lista (I, II, III e IV) de Sintaxe Básica disponível na pasta de exercícios da disciplina disponível no GitHub e entregue na próxima aula.