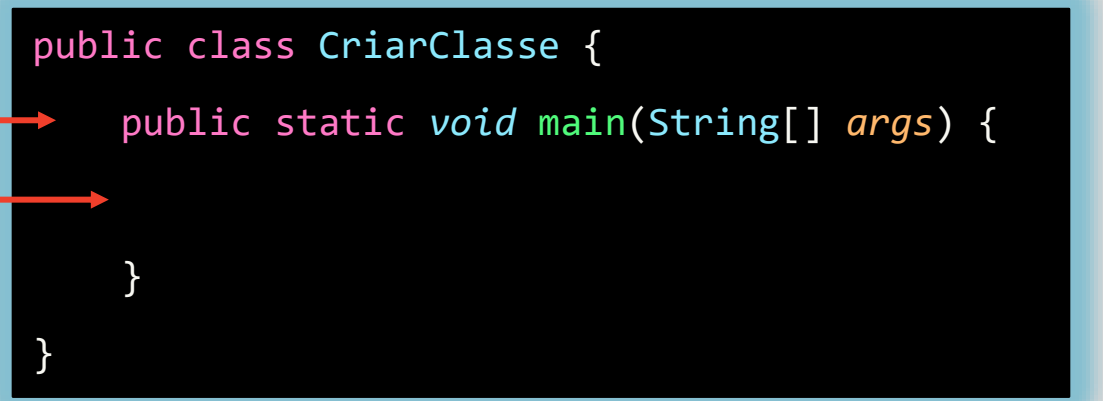
JAVA BÁSICO

Parte 01

Marlison S. da Silva

Estrutura básica de um arquivo Java

- Assinatura da classe
- Assinatura do método
- Local das declarações de variáveis



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

    }
}
```

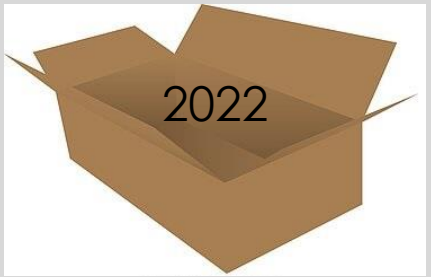
Variáveis

- Guardam dados (informações);

Para criar uma variável é necessário informar:

- Tipo (int), nome da variável (n) e valor dado a ela (0)

Outro exemplo: int ano = 2022;




```java
public class Variavel {

    public static void main(String[] args) {

        int n = 0;

    }

}
```

Variáveis

- Declaração: criação da variável
- Inicialização: dar algum valor pela primeira vez
- Atribuição: dar algum valor (pela primeira vez ou não)

```java
public class Variavel {
    public static void main(String[] args) {
        int n;
        int n = 0; (se variável não estiver declarada)
        n = 0; (se variável estiver declarada)
        n = 100;
    }
}
```

Tipos de dados

```java
public class TiposDados {
    public static void main(String[] args) {
        byte numMuitoCurtoInteiro = 127;
        short numCurtoInteiro = -32000;
        int numInteiro = 1471826438;
        long numLongoInteiro = 471826438123121;
        float numDecimal = 115165312584853.0f;
        double numLongoDecimal = 4812463712789523512400;
        char caracter = '#';
        String texto = "Meu cachorro é muito fofo! ;)";
        boolean verdOuFals = true;
    }
}
```

Tipo	Tamanho	Valor
byte	8 bits	-128 a 127
short	16 bits	-32.768 a 32.767
int	32 bits	-2.147.483.648 a 2.147.483.647
long	64 bits	-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807

Tipo	Tamanho	Valor
float	32 bits	-3.40292347E+38 a +3.40292347E+38
double	64 bits	-1.79769313486231570E+308 a +1.79769313486231570E+308

Tipo	Tamanho	Valor
char	16 bits	'\u0000' a '\uFFFF'

Tipo	Tamanho	Valor
boolean	1 bit	true ou false

Nomes

Variáveis ou métodos (**lowerCamelCase**):

- duasPalavras, _duasPalavras, $duasPalavras;

Nomes de classes (**CamelCase**):

- Classe, DuasPalavras, ComTresPalavras;

Nunca:

- Iniciar com números;
- Conter espaços ou qualquer outro caractere além de letras, números, _ e $;
- Ser um nome pertencente ao Java, como: “class”, “public”, “final”, “int”, “void” e etc.

Sempre:

- Utilizar nomes intuitivos

```
public class ClasseNomes {

    public static void main(String[] args) {

        String duasPalavras = "minusMaiusc";

        String _duasPalavras = "_minusMaiusc";

        String $duasPalavras = "$minusMaiusc";

    }

}
```

Comentários

- São trechos que são ignorados pelo compilador, não sendo executados.

```java
public class Comentario {
    public static void main(String[] args) {
        //Comentário curto relativo a uma linha de código:
        String texto; // Comentário relativo a esta instrução

        /* Comentário mais comprido,
        ocupando uma linha inteira ou várias linhas:
        */

    }
}
```

Imprimir no terminal

- Nesse caso, imprimir é o mesmo que “mostrar”

```java
public class Imprimir {
    public static void main(String[] args) {
        System.out.print("Impressao normal!");
        System.out.println("Impressao que salta uma linha!");
        System.out.print("Impressao normal!");
    }
}
```

Saída (resultado da impressão):

```
C:\Users\marli\Dev\Java\tutoria\slides>java Imprimir
Impressao normal!Impressao que salta uma linha!
Impressao normal!
C:\Users\marli\Dev\Java\tutoria\slides>
```

Como compilar e executar

Compilar:

- Transforma o código em um arquivo que pode ser interpretado: o .class

Executar:

- Interpreta o arquivo .class e executa

Versões antigas
Para compilar:
javac NomeArquivo.java
Para executar:
java NomeArquivo

Novas versões
Executar sem o .class:
java NomeArquivo.java

Exemplo:
Nome do arquivo completo:

- Executar.java

```
C:\Users\marli\Dev\Java\tutoria\slides>javac Executar.java

C:\Users\marli\Dev\Java\tutoria\slides>java Executar
Ola, mundo!

C:\Users\marli\Dev\Java\tutoria\slides>
```

Entrada do teclado - Scanner

Antes de utilizar:

- Necessário importação;
- Necessário instanciar;

Para utilizar:

- Utilizar o nome da variável (leitor), junto do método "next<tipo>()";
  - Exemplo: float -> variavel.nextFloat();
- Exceções:
  - nextLine() para o tipo String
  - next().charAt(0) para tipo char

```java
import java.util.Scanner;
public class UsarScanner {
    public static void main(String[] args) {
        Scanner leitor = new Scanner(System.in);
        String nome;
        nome = leitor.nextLine();
        char simbolo = leitor.next().charAt(0);
    }
}
```

Operadores – atribuição e aritméticos

Atribuição:

- Int telefone **=** 40028922;

Aritméticos:

+	operador de adição (também serve para CONCATENAR)
-	operador subtração
*****	operador de multiplicação
/	operador de divisão
%	operador de módulo (ou resto da divisão)

```java
import java.util.Scanner;
public class Operadores01 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        // atribuindo valores de duas notas
        int nota1 = sc.nextInt(); // 91
        int nota2 = sc.nextInt(); // 100
        // calculando a média
        int mediaInt = (nota1 + nota2)/2; // resultado inteiro
        float mediaDec = (nota1 + nota2)/2f; // resultado decimal
        System.out.println(mediaInt); // 95
        System.out.println(mediaDec); // 95.5
    }
}
```

Operadores – incremento, decremento e igualdade

Decremento: subtrai 1 no valor da variável.

Incremento: adiciona 1 no valor da variável.

Uso:

- Decremento: x--;
- Incremento: x++;

Igual e "não igual":

==	Utilizado quando desejamos verificar se uma variável é igual a outra.
!=	Utilizado quando desejamos verificar se uma variável é diferente de outra.

```
public class Operadores02 {
    public static void main(String[] args) {
        int n = 10;
        n++; // n = n +1 -----> n = 11
        boolean teste = (n == 11);
        System.out.println(teste); // true
        n *= 2; // n = n * 2 -----> n = 22
        teste = (n != 22);
        System.out.println(teste); // false
    }
}
```

Operadores – relacionais e lógicos

Relacionais:

>	Utilizado quando desejamos verificar se uma variável é maior que outra.
>=	Utilizado quando desejamos verificar se uma variável é maior ou igual a outra
<	Utilizado quando desejamos verificar se uma variável é menor que outra.
<=	Utilizado quando desejamos verificar se uma variável é menor ou igual a outra.

Lógicos:

&&	Utilizado quando desejamos que as duas expressões sejam verdadeiras.
||	Utilizado quando precisamos que pelo menos um das expressões seja verdadeira.

```java
public class Operadores03 {
    public static void main(String[] args) {
        int n1 = 11;
        int n2 = 10;

        boolean teste = (n1 == 10 || n2 == 10); //false OU true
        System.out.println(teste); // true
        teste = (n1 == 10 && n2 == 10); // false E true
        System.out.println(teste); // false
    }
}
```

Precedência

Operador	Descrição
`(), (tipo)`	Máxima precedência: separador, conversão de tipo
`+, -, !, ++, --`	Operador unário: positivo, negativo, não (lógico), incremento, decremento
`*, /, %`	Multiplicação, divisão e módulo (inteiros)
`+, -`	Adição, subtração
`<, <=, >=, <`	Operador relacional: menor, menor ou igual, maior ou igual, maior
`==, !=`	Igualdade: igual, diferente
`&&`	Operador lógico **e** condicional
`||`	Operador lógico **ou** condicional
`=`	Atribuição

Condicionais

- Blocos de código que somente serão executados quando algo for verdadeiro

O que podem ser essas condições?

- Qualquer coisa que tenha valor booleano (verdadeiro ou falso)

Uso:

```
if (booleano) {
    bloco de código que será executado
    caso o booleano seja true (verdadeiro)
}
```

```java
public class Condicionais {
    public static void main(String[] args) {
        int n = 1;
        n += 24;
        n *= 4;
        if (n < 100) {
            System.out.println("VERDADEIRO");
        } else {
            System.out.println("FALSO");
        }
    }
}
```

Questão 01

- Faça um programa que receba um char e mostre algo caso o char seja ‘X’.

Questão 02

- Faça um programa para um "jogo" de pontos. O programa já virá com uma palavra guardada. Ao executar o código, o usuário digitará duas palavras e caso a palavra guardada seja igual a uma das duas que o usuário digitou, o programa irá somar 1 ponto. Faça o usuário digitar esses valores por 3 rodadas e mostre os pontos no final de cada rodada.

Referências

- https://www.if.ufrgs.br/~betz/jaulas/aula2o.htm
- https://www.devmedia.com.br/java-operadores-de-atribuicao-aritmeticos-relacionais-e-logicos/38289
- https://www.devmedia.com.br/operadores-logicos-e-matematicos-da-linguagem-java/25248