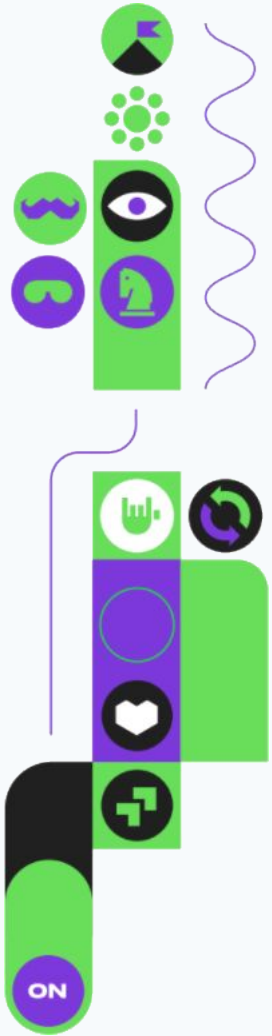


Variáveis & Estruturas de Controles

GAMA ACADEMY

O que é Estrutura de Controle

Git é um sistema de controle de versões distribuído, usado principalmente no desenvolvimento de software para registrar o histórico de edições dos arquivos.

Foi desenvolvido por Linus Torvalds (criador do Linux).

Com o Git podemos desenvolver projetos colaborativos, com diversas pessoas trabalhando simultaneamente no mesmo código sem riscos de perdermos o que fizemos.

O Git guarda um histórico de tudo que foi alterado nos arquivos ao longo do tempo, além de mostrar quem foi o autor da mudança.

Variáveis

Uma variável é uma estrutura que permite que os dados ou expressões sejam armazenados para processamento de informações durante a execução do programa.

Elas variáveis devem ser declaradas antes que possam ser usadas. Declarar uma variável significa criá-la em algum ponto do programa.

A linguagem Java é fortemente tipada. Isso significa que cada variável obrigatoriamente deve ter um tipo declarado quando criada.

Sintaxe: tipo identificador = valor;

Categorias de variáveis:

Variáveis Locais
Podem ser utilizadas dentro do método onde foram declaradas, não sendo acessíveis de outros pontos do programa.

Variáveis de Instância
Uma classe pode conter variáveis que são declaradas fora dos métodos, chamadas de Variáveis de Instância. Seus valores são específicos de cada instância e não são compartilhados entre as instâncias.

Variáveis de Classe
Variáveis declaradas como estáticas são variáveis compartilhadas entre todos os objetos instanciados a partir de uma classe. Por isso, essas variáveis também são conhecidas como Variáveis de Classe.

Tipos de variáveis:

As variáveis de instância de tipo primitivo são inicializadas por padrão, as variáveis dos tipos byte, char, short, int, long, float e double são inicializadas como *0 (zero)*, e as variáveis do tipo boolean são inicializadas como *false*. As variáveis de referência são inicializadas com o valor "null" (nulo).

Tipos primitivos	Tipos por referência
byte	Strings
short	Arrays
int	Objetos
long	...
char	
float	
double	
boolean	

Valores de Armazenamento

boolean	Verdadeiro ou Falso (Valores booleanos)
byte	8 bits
char	16 bits por caractere
double	64 bits
float	32 bits
int	32 bits
string	16 bits por caractere

Estrutura de Controle

Na ciência da computação, uma estrutura de controle (ou fluxo de controle) refere-se à ordem na qual instruções, expressões e chamadas de função são executadas ou avaliadas em um programa de computador sob programação imperativa ou funcional.

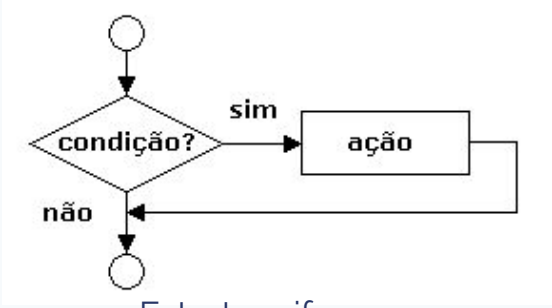
Estrutura de Controle

Estrutura if

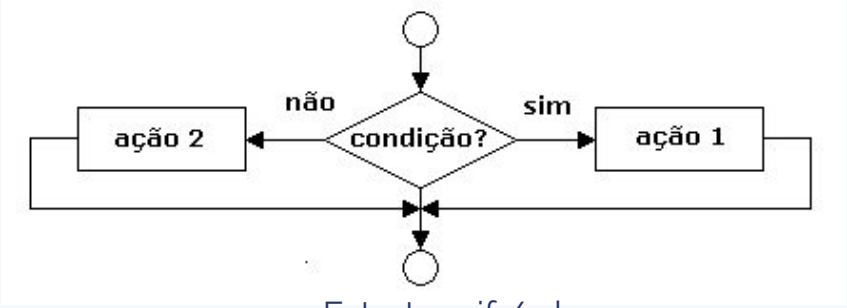
Se *condição* for verdadeira, o programa executa o *bloco de instruções* e contínua para a próxima instrução.

Estrutura if / else

Se *condição* for verdadeira, o programa executa o *bloco de instruções 1*, e se for falsa executa o *bloco de instruções 2*, após o que continua para a próxima instrução.



Estrutura if



Estrutura if / else

Operadores aritméticos

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

Operadores de incremento e decremento

numero++;

numeros--;

Operadores de igualdade

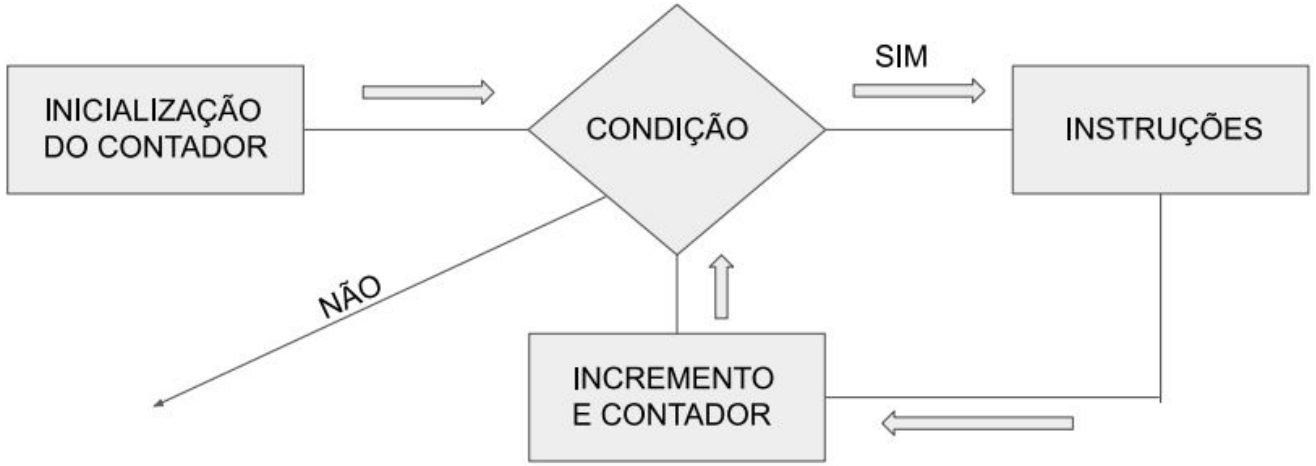
igual(“==”) ou diferente(“!=”)

Operadores de operações relacionais

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

Estrutura de repetição

Dentro da lógica de programação é uma **estrutura** que permite executar mais de uma vez o mesmo comando ou conjunto de comandos, de acordo com uma condição ou com um contador.



Estrutura de repetição

O que são?

- São comandos que permitem que uma sequência de instruções seja executada várias vezes até que uma condição seja satisfeita;

- Se uma instrução ou uma sequência de instruções precisa ser executada várias vezes, deve-se utilizar uma estrutura de repetição.

Para que servem?

- Servem para repetir um conjunto de instruções sem que seja necessário escrevê-las várias vezes;

- Permitem que um trecho do algoritmo seja repetido, em um número determinado ou indeterminado de vezes, sem que o código a ser repetido tenha que ser escrito novamente;

- As estruturas de repetição também são chamadas de Laços ou Loops.

Estrutura de repetição

Funcionamento

- As estruturas de repetição envolvem a avaliação de uma condição (teste);
- A avaliação resulta em valores Verdadeiros ou Falsos;
- Se o resultado da condição é Falso, não é iniciada a repetição ou, caso esteja em execução, é encerrada a repetição;
- Se o resultado da condição for Verdadeiro, é iniciada a repetição ou, caso esteja em execução, é reiniciada a execução das instruções dentro da Estrutura de Repetição;
- A avaliação da condição é sempre novamente realizada após a execução da última instrução dentro da estrutura de repetição;
- A única Estrutura de Repetição que não realiza a avaliação da condição antes de iniciar é a Do/While (Faça/Enquanto).
- Desta forma, é assegurado que todas as instruções dentro da Estrutura de Repetição do Do/While serão executadas pelo menos uma vez.

Estrutura de repetição

Tipos de Estruturas de Repetição

While

O termo **while** pode ser traduzido para o português como “enquanto”. Este termo é utilizado para construir uma estrutura de repetição que executa, repetidamente, uma única instrução ou um bloco delas “enquanto” uma expressão booleana for verdadeira.

```
//INCREMENTADO - de 0 à 9
        int i = 0;
        while(i<=9){
            i = i + 1;
            System.out.println( i );
        }
```

Estrutura de repetição

Do While

A diferença desse iterador para os outros, é que o bloco de instrução será executado no mínimo uma única vez. Como podemos ver no exemplo abaixo:

```
int i = 0;

        do{
            System.out.println( i );
            i++;
        }while(i <= 10);
```

Estrutura de repetição

FOR

Controlando fluxo com laços

```
for (int i = 0; i < 5; i++) {
 if( i % 2 ==  0) {
  System.out.println(i);
 }
}
```

For Each

Projetado especificamente para iterar sobre matrizes e coleção de objetos.

```
String[] cars = {"Volvo", "BMW", "Ford", "Mazda"};

for (String i : cars) {
  System.out.println(i);
}
```

Atividade em Grupo

Obrigada

GAMA ACADEMY