**Anotações – Java 2024**

Auxilio: Curso mais didático e completo de Java e OO, UML, JDBC, JavaFX, Spring Boot, JPA, Hibernate, MySQL, MongoDB e muito mais.

Plataforma: Udemy

Professor: Nelio Alves

Algoritmo

Sequência finita de instruções para se resolver um problema.

Exemplo:

Problema: lavar roupa suja

Algoritmo:

1. Colocar a roupa em um recipiente;
2. Colocar um pouco de sabão e amaciante;
3. Encher de água;
4. Mexer tudo até dissolver todo o sabão;
5. Deixar de molho por vinte minutos;
6. Esfregar a roupa;
7. Enxaguar;
8. Torcer.

Automação

Consiste em utilizar máquina(s) para executar o procedimento desejado de forma automática ou semiautomática.

Computador

Hardware – parte física (a máquina em si)

Software – parte lógica (programas)

* Sistema operacional (Windows, Linux, iOS);
* Aplicativos (apps de escritório, app de câmera, navegador web);
* Jogos;
* Utilitários (Antivírus, compactador de arquivos);
* Outros.

Programa > Algoritmo

Programas de computador **são algoritmos** executados pelo computador (em linhas gerais).

Conclusão: o computador é uma máquina de **automatiza** a execução de **algoritmos**.

Qualquer algoritmo? Não. Apenas algoritmos computacionais:

* Processamento de dados;
* Cálculos.

Linguagem de programação, léxica, sintática.

Linguagem de programação:

É um conjunto de regras **léxicas** (ortografia) e **sintáticas** (gramática) para se escrever programas.

Léxica

Diz respeito à correção das palavras “isoladas” (ortografia).

Exemplo (Português):

* cachorro – correto
* caxorro – errado

Exemplo (Linguagem de programação):

* main – correto
* maim – errado

Sintática

Diz respeito à correção das **sentenças** (gramática).

Exemplo (Português):

* O cachorro está com fome - certo
* A cachorro está com fome – errado

Exemplo (Linguagem de programação):

* x = 2 + y; - certo
* x = + 2 y; - errado

Exemplo de um programa:

Suponha um programa que solicita do usuário dois números e depois mostra na média aritmética deles:

*Digite o primeiro numero: 3*

*Digite o segundo numero: 6*

*Media = 4.5*

Solução em linguagem C

*#include <stdio.h>*

*int main() {*

*double x, y, media;*

*printf(“Digite o primeiro numero: “);*

*scanf(“%1f”, &x);*

*printf(“Digite o segundo numero: “);*

*scanf(“%1f”, &y);*

*media = (x + y) / 2.0;*

*printf(“Media = %.1f\n”, media);*

*return 0 ;*

*}*

Solução em linguagem C++

*#include <iostream>*

*using namespace std;*

*int main() {*

*double x, y, media;*

*cout << “Digite o primeiro numero: “;*

*cout >> x;*

*cin >> y;*

*media = (x + y) / 2.0;*

*cout << “Media = “ << media << endl;*

*return 0;*

*}*

Solução em linguagem C#

*using System;*

*namespace programa {*

*class Program {*

*static void Main(string[] args) {*

*double x, y, media;*

*Console.Write(“Digite o primeiro numero: “);*

*x = double.Parse(Console.ReadLine());*

*y = double.Parse(Console.ReadLine());*

*media = (x + y) / 2.0;*

*Console.WriteLine(“Media = “ + media);*

*}*

*}*

*}*

Solução em linguagem Java

*import java.util.Scanner;*

*public class Main {*

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

*Scanner sc = new Scanner(System.in);*

*double x, y, media;*

*System.out.print(“Digite o primeiro numero: “);*

*x = sc.nextDouble();*

*System.out.print(“Digite o segundo numero: “);*

*y = sc.nextDouble();*

*media = (x + y) / 2.0;*

*System.out.println(“Media = “ + media);*

*sc.close();*

*}*

*}*

**IDE – Ambiente Integrado de Desenvolvimento**

É um conjunto de softwares utilizado para a construção de programas.

Exemplos:

* C/C++ : Code Blocks
* Java: Eclipse, NetBeans
* C#: Microsoft Visual Studio

Funcionalidades de uma IDE

* Edição de código fonte (endentação, autocompletar, destaque de palavras, etc.);
* Depuração e testes;
* Construção do produto final (build);
* Sugestão de modelos (templates);
* Auxiliar em várias tarefas do seu projeto;
* Etc.

Compilação e interpretação – Código fonte e objeto Máquina virtual

Código fonte: é aquele escrito pelo programador em linguagem de programação

Compilação

Código fonte -> **compilador: compilação** | *análise léxica + análise sintática* -> Código objeto -> **Gerador de código: construção (build)** | *geração de código* -> Código executável -> **Execução**

**Exemplos de linguagem que tipicamente usam essa abordagem: C, C++**

Interpretação

Código fonte -> **Interpretador: interpretação** | *análise léxica + análise sintática + geração de código SOB DEMANDA* -> **Execução**

Abordagem híbrida

Código fonte **-> Compilador: precompilação** | *análise léxica + análise sintática* -> Bytecode -> **Máquina virtual: interpretação** | *geração de código SOB DEMANDA* -> Execução

**Exemplos de linguagem que tipicamente usam essa abordagem: Java (JVM), C# (Microsoft .NET Framework)**

**Vantagens**

**Compilação:**

* **velocidade do programa**
* **auxílio do compilador antes da execução**

**Interpretação:**

* **flexibilidade de manutenção do aplicativo em produção**
* **expressividade da linguagem**
* **código fonte não precisa ser recompilado para rodar em plataformas diferentes**

**Abordagem híbrida:**

* velocidade do programa
* **auxílio do compilador antes da execução**
* flexibilidade de manutenção do aplicativo em produção
* **código fonte não precisa ser recompilado para rodar em plataformas diferentes**

**C/C++**

Código fonte

Código executável (específico para o sistema operacional)

Sistema Operacional

Hardware

**PHP, Python, JavaScript**

Código fonte

Interpretador (específico para o sistema operacional)

Sistema Operacional

Hardware

**Java, C#**

Código fonte

Bytecode (código precompilado)

Máquina virtual (específica para o sistema operacional)

Sistema Operacional

Hardware

**Java – contextualização**

O que é Java?

Linguagem de programação (regras sintáticas)

Plataforma de desenvolvimento e execução

* Bibliotecas (API)
* Ambientes de execução

Histórico

Problemas resolvidos e motivo do seu sucesso:

* Ponteiros / gerenciamento de memória
* Portabilidade falha: reescrever parte do código ao mudar de SO
* Utilização em dispositivos diversos
* Custo

Criada pela Sun Microsystems no meio da década de 1990

Adquirida pela Oracle Corporation em 2010.

**Aspectos notáveis**

* Código compilado para bytecode e executado em máquina virtual (JVM)
* Portável, segura, robusta
* Roda em vários tipos de dispositivos
* Domina o mercado corporativo desde o fim do século 20
* Padrão Android por muitos anos

**Edições**

* Java ME – Java Micro Edition – dispositivos embarcados e móveis – IoT
* <http://www.oracle.com/technetwork/java/javame>
* Java SE – Java Standard Edition – core – desktop e servidores
* <http://www.oracle.com/technetwork/java/javase>
* Java EE – Java Enterprise Edition – aplicações corporativas
* <http://www.oracle.com/technetwork/java/javaee>

**Plataforma Java SE**

**JVM – Java Virtual Machine**

* Máquina virtual do Java – necessário para executar sistemas Java

**Compilação e interpretação**

* Linguagens **compiladas**: C, C++
* Linguagens **interpretadas**: PHP, JavaScript
* Linguagens **pré-compiladas + máquina virtual**: Java, C#

**Estrutura de uma aplicação Java**

Uma aplicação é composta por classes.

package = agrupamento LÓGICO de classes relacionadas.

Módulo (Java 9+) = Agrupamento lógico de pacotes relacionados.

Runtime = Agrupamento físico.

Aplicação = Agrupamento de módulo relacionados.

**Expressões aritméticas**

Expressão – resultado -> valor numérico

4 + 5 resultado -> 9

**Operadores aritméticos**

*C, C++, Java, C#*

**Operador Significado**

+ adição

- subtração

\* multiplicação

/ divisão

% resto da divisão (“mod”)

**Precedência**: 1º lugar: \* / % 2º lugar: + -

Exemplos de expressões aritméticas

2 \* 6 / 3 Resultado = 4

3 + 2 \* 4 Resultado = 11

(3 + 2) \* 4 Resultado = 20

60 / (3 + 2) \* 4 Resultado = 48

60 / ((3 + 2) \* 4) Resultado = 3

Exemplos com operador “mod”

14 % 3 Resultado = 2

19 % 5 Resultado = 4

**Variáveis - Tipos básicos em Java**

Um programa de computador em execução lida com dados.

Como esses dados são armazenados?

* Em **variáveis**!

**Variáveis**

Definição informal:

Em programação, uma variável é uma porção de memória (RAM) utilizada para armazenar dados durante a execução dos programas.

Computador > Memória RAM > x = 3; salario = 5000.0; nome = “Maria”

**Declaração de variáveis**

**Sintaxe:**

<tipo> <nome> = <valor inicial>;

Valor inicial é opcional.

**Exemplos:**

int idade = 25;

double altura = 1.68;

char sexp = ‘F’;

*Memória RAM:*

idade = 25

altura = 1.68

sexo = F

**Uma variável possui:**

* Nome (ou identificador)
* Tipo
* Valor
* Endereço

Tipos primitivos em Java



Um bit pode armazenar 2 valores possíveis (0 ou 1)

Cada bit = 2 possibilidades

8 bits:

2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 = 28 = 256 possibilidades

**Nomes de variáveis**

* Não pode começar com dígito: use uma letra ou \_
* Não pode ter espaço em branco
* Não usar acentos ou til
* Sugestão: use o padrão “camel case”

**Errado: Correto:**

*int 5minutos; int \_5minutos;*

*int salário; int salario;*

*int salário do funcionário; int salarioDoFuncionario;*

**As três operações básicas de programação**

Um programa de computador é capaz de realizar essencialmente três operações:

* Entrada de dados;
* Processamento de dados;
* Saída de dados.

**Entrada de dados:**

Usuário -> Programa (dentro de variáveis). Também chamada de LEITURA: “O programa está lendo dados.”

**Processamento de dados:**

É quando o programa realiza os cálculos. O processamento de dados se dá por um comando chamado ATRIBUIÇÃO. media = ( x + y ) / 2.0;

**Saída de dados:**

Programa -> Usuário. Também chamada de ESCRITA: “O programa está escrevendo dados.”

**Saída de dados em Java:**

Para escrever na tela um texto qualquer

**Sem quebra de linha ao final:**

System.out.print(“Bom dia!”);

**Com quebra de linha ao final:**

System.out.println(“Bom dia!”);

Para escrever o conteúdo de uma variável de algum tipo básico:

Suponha uma variável tipo int declarada e iniciada:

*int y = 32;*

*System.out.println(y);*

Para escrever o conteúdo de uma variável com ponto flutuante:

Suponha uma variável tipo double declarada e iniciada:

*double x = 10.35784;*

%n = quebra de linha (independente de plataforma)

*System.out.println(x);*

*System.out.printf(“%.2f%n”, x); - System.out.printf(“%.4f%n”, x);* - Localidade do sistema

**ATENÇÃO:**

Para considerar o separador de decimais como ponto, ANTES da declaração do Scanner, faça:

*Locale.setDefault(Locale.US);*

Para concatenar vários elementos em um mesmo comando de escrita:  
Regra geral para print e println:

elemento1 + elemento2 + elemento3 + ... + elementoN

*System.out.println(“RESULTADO = “* ***+*** *x* ***+*** *“ METROS”);*

Para concatenar vários elementos em um mesmo comando de escrita:

Regra geral para **printf**:

“TEXTO1 %f TEXTO2 %f TEXT03”, variavel1, variavel2

%f = ponto flutuante

%n = quebra de linha

*System.out.printf(“RESULTADO = %.2f metros%n”, x);*

%f = ponto flutuante

%d = inteiro

%s = texto

%n = quebra de linha

*String nome = “Maria”;*

*int idade = 31;*

*double renda = 4000.0;*

*System.out.printf(“%s tem %d anos e ganha R$ %.2f reais%n”, nome, idade, renda);*

*Mais informações: https://docs.oracle.com/javase/tutorial/java/data/numberformat.html*

**Processamento de dados em Java, Casting**

Processamento de dados

Comando de atribuição.

**Sintaxe:**

<variável> = <expressão>;

= Lê-se “recebe”

**Regra:**

1. A expressão é calculada;
2. O resultado da expressão é armazenado na variável.

**Exemplo 1:**

*int x, y;*

*x = 5;*

*y = 2 \* x;*

*System.out.println(x);*

*System.out.println(y);*

**Exemplo 2:**

*int x;*

*double y;*

*x = 5;*

*y = 2 \* x;*

*System.out.println(x);*

*System.out.println(y);*

**Exemplo 3:**

*double b, B, h, area;*

*b = 6.0;*

*B = 8.0;*

*h = 5.0;*

*area = (b + B) / 2.0 \* h;*

*System.out.println(area);*

**Boa prática:**

Sempre indique o tipo do número, se a expressão for de ponto flutuante (não inteira).

Para **double** use:

*.0*

Para **float** use:

*f*

**Exemplo:**

***float*** *b, B, h, area;*

*b = 6f;*

*B = 8f;*

*h = 5f;*

*area = (b + B) / 2f \* h;*

*System.out.println(area);*

**Exemplo 4:**

*int a, g;*

*double resultado;*

*a = 5;*

*g = 2;*

*resultado = (double) a / b;* ***-> casting***

*System.out.println(resultado);*

**Exemplo 5:**

*double j;*

*int p;*

*j = 5.0;*

*p = (int) j;* ***-> casting***

*System.out.println(p);*

Entrada de Dados

**Usuário** -> **Programa** (dentro de variáveis)

Dispositivo de ENTRADA (teclado)

*Também chamada de LEITURA:*

*"O programa está lendo dados."*

**Scanner**

Para fazer entrada de dados, nós vamos criar um objeto do tipo "Scanner" da seguinte forma:

*Scanner sc = new Scanner(System.in);*

**E para funcionar precisamos utilizar:**

*import java.util.Scanner;*

faça sc.close() quando não precisar mais do objeto sc

**Para ler uma palavra (texto sem espaços)**

Suponha uma variável tipo String declarada:

*String x;*

*x = sc.next();*

Exemplo 01:

package entradaDeDados;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner keyboard = new Scanner(System.***in***);

String nome;

nome = keyboard.next();

System.***out***.println("Voce digitou: " + nome);

keyboard.close();

}

}

**Para ler um número inteiro**

Suponha uma variável tipo int declarada:

*int x;*

*x = keyboard.nextInt();*

Exemplo 02:

package entradaDeDados;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner keyboard = new Scanner(System.***in***);

String nome;

int idade;

System.***out***.println("Digite o seu nome:");

nome = keyboard.next();

System.***out***.println("Voce digitou: " + nome);

System.***out***.println("Digite a sua idade:");

idade = keyboard.nextInt();

System.***out***.println("Voce digitou: " + idade);

keyboard.close();

}

}

**Para ler um número com ponto flutuante**

Suponha uma variável tipo double declarada:

*double x;*

*x = sc.nextDouble(); -> Localidade do sistema*

ATENÇÃO:

Para considerar o separador de decimais como ponto, ANTES da declaração do Scanner, faça:

*Locale.setDefault(Locale.US);*

Exemplo 03:

package entradaDeDados;

import java.util.Locale;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Locale.*setDefault*(Locale.***US***);

Scanner keyboard = new Scanner(System.***in***);

String nome;

int idade;

double altura;

System.***out***.println("Digite o seu nome:");

nome = keyboard.next();

System.***out***.println("Voce digitou: " + nome);

System.***out***.println("\nDigite a sua idade:");

idade = keyboard.nextInt();

System.***out***.println("Voce digitou: " + idade);

System.***out***.println("\nDigite a sua altura:");

altura = keyboard.nextDouble();

System.***out***.printf("Voce digitou: %.2f%n", altura);

keyboard.close();

}

}

Locale precisa ser feito antes do Scanner para conseguir receber entrada de dados com . Ao invés de virgula.

Se não quiser usar . e sim virgula, basta não usar o Locale.

**Para ler um caractere**

Suponha uma variável char declarada:

*char x;*

*x = sc.next().charAt(0);*

**Para ler vários dados na mesma linha**

*string x;*

*int y;*

*double z;*

*x = sc.next();*

*y = sc;nextInt();*

*z = sc.nextDouble();*

Para ler um texto ATÉ A QUEBRA DE LINHA

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String s1, s2, s3;

s1 = sc.nextLine();

s2 = sc.nextLine();

s3 = sc.nextLine();

System.out.println(“Dados Digitados:”);

System.out.println(s1);

System.out.println(s2);

System.out.println(s3);

sc.close();

}

}

**ATENÇÃO: quebra de linha pendente**

int x;

String s1, s2, s3;

x = sc.nextInt();

s1 = sc.nextLine();

s2 = sc.nextLine();

s3 = sc.nextLine();

System.out.println(“DADOS DIGITADOS:”);

System.out.println(x);

System.out.println(s1);

System.out.println(s2);

System.out.println(s3);

Quando você usa um comado de leitura diferente do nextLine() e dá alguma quebra de linha, essa quebra de linha fica “pendente” na entrada padrão.

Se você então fizer um nextLine(), aquela quebra de linha pendente será absorvida pelo nextLine().

**Solução:**

Faça um nextLine() extra antes de fazer o nextLine() do seu interesse.

**Exemplo**:

package entradaDeDados02;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.***in***);

int x;

String s1, s2, s3;

x = sc.nextInt();

sc.nextLine();

s1 = sc.nextLine();

s2 = sc.nextLine();

s3 = sc.nextLine();

System.***out***.println("Dados Digitados:");

System.***out***.println(x);

System.***out***.println(s1);

System.***out***.println(s2);

System.***out***.println(s3);

sc.close();

}

}

Resumo da aula:

**Scanner**

* next()
* nextInt()
* nextDouble()
* next().charAt(0)

**Locale**

**Como ler até a quebra de linha**

* nextLine()
* como limpar o buffer de leitura

**Funções matemáticas:**

Algumas funções matemáticas em Java

**Exemplo: Significado**

A = Math.sqrt(x); Variável A recebe a raiz quadrada de x

A = Math.pow(x, y); Variável A recebe o resultado de x elevado a y

A = Math.abs(x); Variável A recebe o valor absoluto de x

Exemplo:

package funcoesMatematica;

public class Application {

public static void main(String[] args) {

double x = 3.0;

double y = 4.0;

double z = -5.0;

double A, B, C;

A = Math.*sqrt*(x);

B = Math.*sqrt*(y);

C = Math.*sqrt*(25.0);

System.***out***.println("Raiz quadrada de " + x + " = " + A);

System.***out***.println("Raiz quadrada de " + y + " = " + B);

System.***out***.println("Raiz quadrada de 25 = " + C);

A = Math.*pow*(x, y);

B = Math.*pow*(x, 2.0);

C = Math.*pow*(5.0, 2.0);

System.***out***.println(x + " elevado a " + y + " = " + A);

System.***out***.println(x + " elevado ao quadrado = " + B);

System.***out***.println("5 elevado ao quadrado = " + C);

A = Math.*abs*(y);

B = Math.*abs*(z);

System.***out***.println("Valor absoluto de " + y + " = " + A);

System.***out***.println("Valor absoluto de " + z + " = " + B);

}

}

**Incluindo funções em expressões maiores**

*x = -b +- raiz de delta / 2.a*

delta = b² -4ac

delta = Math.pow(b, 2.0) – 4\*a\*c;

x1 = (-b + Math.sqrt(delta)) / (2.0 \* a);

x2 = (-b – Math.sqrt(delta)) / (2.0 \* a);

**Exercício resolvido 01:**

Fazer um programa para ler as medidas da largura e comprimento de um terreno retangular com uma casa decimal, bem como o valor do metro quadrado do terreno com duas casas decimais. Em seguida, o programa deve mostrar o valor da área do terreno, bem como o valor do preço do terreno, ambos com duas casas decimais, conforme exemplo.

Exemplo:

Entrada: Saída:

10.0 AREA = 300.00

30.0 PRECO = 60000.00

200.00

área = largura x comprimento

preço = área x preço do metro quadrado

**Exercícios propostos (No arquivo: 02-exercicios1-estrutura-sequencial.pdf)**

**Exercício 01**

*Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1003.java*

Faça um programa para ler dois valores inteiros, e depois mostrar na tela a soma desses números com uma mensagem explicativa, conforme exemplos.

**Exemplos:**

**Entrada:**

10

30

**Saída:**

SOMA = 40

**Entrada:**

-30

10

**Saída:**

SOMA = -20

**Entrada:**

0

0

**Saída:**

SOMA = 0

**Exercício 02**

*Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1002.java*

Faça um programa para ler o valor do raio de um círculo, e depois mostrar o valor da área deste círculo com quatro

casas decimais conforme exemplos.

Fórmula da área: area = π . raio2

Considere o valor de π = 3.14159

**Exemplos:**

**Entrada:**

2.00

**Saída:**

A=12.5664

**Entrada:**

100.64

**Saída:**

A=31819.3103

**Entrada:**

150.00

**Saída:**

A=70685.7750

**Exercício 03**

*Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1007.java*

Fazer um programa para ler quatro valores inteiros A, B, C e D. A seguir, calcule e mostre a diferença do produto

de A e B pelo produto de C e D segundo a fórmula: DIFERENCA = (A \* B - C \* D).

**Exemplos:**

**Entrada:**

5

6

7

8

**Saída:**

DIFERENCA = -26

**Entrada:**

5

6

-7

8

**Saída:**

DIFERENCA = 86

**Exercício 04**

*Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1008.java*

Fazer um programa que leia o número de um funcionário, seu número de horas trabalhadas, o valor que recebe por

hora e calcula o salário desse funcionário. A seguir, mostre o número e o salário do funcionário, com duas casas

decimais.

Exemplos:

**Entrada:**

25

100

5.50

**Saída:**

NUMBER = 25

SALARY = U$ 550.00

**Entrada:**

1

200

20.50

**Saída:**

NUMBER = 1

SALARY = U$ 4100.00

**Entrada:**

6

145

15.55

**Saída:**

NUMBER = 6

SALARY = U$ 2254.75

**Exercício 05**

*Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1010.java*

Fazer um programa para ler o código de uma peça 1, o número de peças 1, o valor unitário de cada peça 1, o

código de uma peça 2, o número de peças 2 e o valor unitário de cada peça 2. Calcule e mostre o valor a ser pago.

Exemplos:

**Entrada:**

12 1 5.30

16 2 5.10

**Saída:**

VALOR A PAGAR: R$ 15.50

**Entrada:**

13 2 15.30

161 4 5.20

**Saída:**

VALOR A PAGAR: R$ 51.40

**Entrada:**

1 1 15.10

2 1 15.10

**Saída:**

VALOR A PAGAR: R$ 30.20

**Exercício 06**

Correção: https://github.com/acenelio/nivelamento-java/blob/master/src/uri1012.java

Fazer um programa que leia três valores com ponto flutuante de dupla precisão: A, B e C. Em seguida, calcule e

mostre:

a) a área do triângulo retângulo que tem A por base e C por altura.

b) a área do círculo de raio C. (pi = 3.14159)

c) a área do trapézio que tem A e B por bases e C por altura.

d) a área do quadrado que tem lado B.

e) a área do retângulo que tem lados A e B.

Exemplos:

**Entrada:**

3.0 4.0 5.2

**Saída:**

TRIANGULO: 7.800

CIRCULO: 84.949

TRAPEZIO: 18.200

QUADRADO: 16.000

RETANGULO: 12.000

**Entrada:**

12.7 10.4 15.2

**Saída:**

TRIANGULO: 96.520

CIRCULO: 725.833

TRAPEZIO: 175.560

QUADRADO: 108.160

RETANGULO: 132.080

**Expressões comparativas**

expressão -> resultado -> valor verdade

5 > 10 -> Falso

**Operadores comparativos**

C, C++, Java, C#

**Operador Significado**

> maior

< menor

>= maior ou igual

<= menor ou igual

== igual

!= diferente

**Exemplos de expressões comparativas**

(*suponha x igual a 5*)

X > 0 Resultado: V

X == 3 Resultado: F

10 <= 30 Resultado: V

X != 2 Resultado: V

**Expressões lógicas**

expressão -> resultado -> valor verdade

**Operadores lógicos**

C, C++, Java, C#

Operador Significado

&& E

|| OU

! NÃO

***Ideia por trás do operador "E"***

Você pode obter uma habilitação de motorista se:

* For aprovado no exame psicotécnico,

**E**

* For aprovado no exame de legislação,

**E**

* For aprovado no exame de direção

*Todas as condições devem ser verdadeiras!*

**Exemplos de expressões lógicas**

(*suponha x igual a 5*):



Tabela verdade do operador "**E**"

**A B A&&B**

F F F

F V F

V F F

V V V

**Ideia por trás do operador "OU"**

Você pode obter estacionar na vaga especial se:

* For idoso(a),

**OU**

* For uma pessoa com deficiência,

**OU**

* For uma gestante

*Pelo menos uma condição deve ser verdadeira!*

**Exemplos de expressões lógicas**

(*suponha x igual a 5*)



Tabela verdade do operador "**OU**"

**A B A || B**

F F F

F V V

V F V

V V V

**Ideia por trás do operador "NÃO"**

Você tem direito a receber uma bolsa de estudos se você:

**Não**

* Possuir renda maior que $ 3000,00

*O operador "NÃO" inverte a condição*

**Exemplos de expressões lógicas**

(*suponha x igual a 5*)

Tabela verdade do operador "**NÃO**"

**A !A**

F V

V F

**Estrutura condicional**

**Conceito**

Estrutura condicional: É uma **estrutura de controle** que permite definir que um certo **bloco de comandos** somente será executado dependendo de uma **condição**.



**Sintaxe da estrutura condicional**

**Simples**:

if (<condição>) {

<comando 1>

<comando 2>

}

**Exemplo**:

public class Main {

public static void main(String[] args) {

int x = 5;

System.out.println("Bom dia");

if (x < 0){

System.out.println("Boa tarde");

}

System.out.println("Boa noite");

}

}

**Resultado:**

*Bom dia*

*Boa noite*

**REGRA:**

**V**: *executa o bloco de comandos*

**F**: *pula o bloco de comandos*

*Sintaxe da estrutura condicional*

**Composta**:

if (<condição>){

<comando 1>

<comando 2>

}

else {

<comando 3>

<comando 4>

}

**REGRA**:

**V**: *executa somente o bloco do if*

**F**: *executa somente o bloco do eles*

**Exemplo:**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner keyboard = new Scanner(System.in);

int hora;

System.out.println("Quantas horas?");

hora = keyboard.nextInt();

if (hora < 12){

System.out.println("Bom dia");

}

else {

System.out.println("Boa tarde");

}

keyboard.close();

}

}

**Resultado:**

10

Bom dia

**Resultado:**

15

Boa tarde

E se tiver mais de duas possibilidades:

horas < 12 Bom dia

12 <= horas < 18 Boa tarde

horas >= 18 Boa noite

**Encadeamento de estruturas condicionais**

**if** (condição 1) {

comando 1

comando 2

}

**else** {

**if** (condição 2) {

comando 3

comando 4

}

**else** {

comando 5

comando 6

}

}

**Exemplo:**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner keyboard = new Scanner(System.in);

int hora;

System.out.println("Quantas horas?");

hora = keyboard.nextInt();

if (hora < 12){

System.out.println("Bom dia");

}

else if (hora < 18) {

System.out.println("Boa tarde");

}

else {

System.out.println("Boa noite");

}

keyboard.close();

}

}

**Encademaneto de estruturas condicionais**

if (condição 1) {

comando 1

comando 2

}

else if (condição 2) {

comando 3

comando 4

}

else if (condição 2) {

comando 5

comando 6

}

else {

comando 7

comando 8

}

**Sintaxe opcional: operadores de atribuição cumulativa**

Problema exemplo

Uma operadora de telefonia cobra R$ 50.00 por um plano básico que dá direito a 100 minutos de telefone. Cada minuto que exceder a franquia de 100 minutos custa R 2.00. Fazer um programa para ler a quantidade de minutos que uma pessoa consumiu, daí mostrar o valor a ser pago.

Entrada:

22

Saída:

Valor a pagar: R$ 50.00

Entrada:

103

Saída:

Valor a pagar: R$ 56.00

**Operadores de atribuição cumulativa:**

a += b; a = a + b;

a -= b; a = a – b;

a \* = b; a = a \* b;

a /= b; a = a / b;

a %= b; a = a % b;

**Sintaxe opcional: estrutura switch-case**

Quando se tem várias opções de fluxo a serem tratadas com base no valor de uma variável, ao invés de várias estruturas if-else encadeadas, alguns preferem utilizar a estrutura switch-case.

Problema exemplo

Fazer um programa para ler um valor inteiro de 1 a 7 representado um dia da semana (sendo 1 = domingo, 2 = segunda, e assim por diante).

Escrever na tela o dia da semana correspondente, conforme exemplos.

**Entrada Saída**

1 Dia da semana:domingo

4 Dia da semana:quarta

9 Dia da semana:valor inválido

**Sintaxe**:

*switch ( expressão ) {*

*case valor1:*

*comando1*

*comando2*

*break;*

*case valor2:*

*comando3*

*comando4*

*break;*

*default:*

*comando5*

*comando6*

*break;*

*}*

**Expressão condicional ternária**

Estrutura opcional ao if-else quando se deseja decidir um VALOR com base em uma condição.

**Sintaxe:**

( condição ) ? valor\_se\_verdadeiro : valor\_se\_falso

**Exemplos:**

( 2 > 4 ) ? 50 : 80 -> 80

( 10 != 3 ) ? “Maria” : “Alex” -> “Maria”

**Demonstração:**

double preco = 34.5;

double desconto;

if (preco < 20.0) {

desconto = preco \* 0.1;

} else {

desconto = preco \* 0.05;

}

**Condicional ternária (Código):**

double preco = 34.5;

double desconto = (preco < 20.0) ? preco \* 0.1 : preco \* 0.05;

**Escopo e inicialização**

* Escopo de uma variável: é a região do programa onde a variável é válida, ou seja, onde ela pode ser referenciada;
* Uma variável não pode ser usada se não for iniciada;
* Escopo de métodos no Capítulo 5.

Inicialização – Uma variável deve ser inicializada de alguma forma, antes de apresentar a saída.   
Deve conter algum valor.

Escopo – A variável não pode ficar fora do escopo. Exemplo:



discount está fora do escopo, no caso a declaração feita da variável está dentro do if, não sendo possível localizar o resultado fora dele.

Mesma coisa se a variável discount estivesse sido declarada antes, porém, só tivesse um valor atribuído dentro do if. E ao tentar apresentar no println fora da estrutura condicional, o programa iria apresentar erro. Pois o valor só foi atribuído dentro do if e caso não fosse verdadeiro, não iria processar o resultado do if. Nesse caso, precisamos atribuir algum valor antes, na variável discount para poder obter o resultado (ou utilizar um else com valor zero na variável)

Exemplo:



**Como utilizar o Debug no Eclipse (execução passo a passo)**

**Para marcar uma linha de breakpoint:**

* Run -> Toggle Breakpoint

**Para iniciar o debug:**

* Botão direito na classe -> Debug as -> Java Application

**Para executar uma linha:**

* F6

**Para interromper o debug:**

* **Botão stop**

**While**

Estrutura repetitiva “enquanto” é uma **estrutura de controle** que **repete** um bloco de comandos enquanto uma **condição** for verdadeira. **Quando usar**: quando **não** se sabe previamente a quantidade de repetições que será realizada.

Problema exemplo:

Fazer um programa que lê números inteiros até que um zero seja lido. Ao final mostra a soma dos números lidos.

Entrada: Saída:

5 11

2

4

0

Sintaxe / regra:

While ( condição ) {

comando 1

comando 2

}

**Regra**:

**V**: executa e volta

**F**: pula fora

**Estrutura repetitiva “para” (for)**

É uma estrutura de controle que repete um bloco de comandos para um certo intervalo de valores.

Quando usar: quando se sabe previamente a quantidade de repetições, ou o intervalo de valores.

Por exemplo:

Fazer um programa que lê um valor inteiro N e depois N números inteiros.

Ao final, mostra a soma dos N números lidos:

Entrada: Saída:

3 11

5

2

4

Sintaxe/regra:



Importante

Perceba que a estrutura “para” é ótima para se fazer uma repetição baseada em uma CONTAGEM:



Estrutura repetitiva “faça-enquanto” – Do While

**Sintaxe/regra**

***Do*** *{*

*Comando1*

*Comando2*

*}* ***while*** *(condição);*

**Regra:**

V: volta

F: pula fora

**Problema exemplo:**

Fazer um programa para ler uma temperatura em Celsius e mostrar o equivalente em Fahrenheit. Perguntar se o usuário deseja repetir (s/n). Caso o usuário digite “s”, repetir o programa.

Fórmula: F = 9C/5 + 32

**Exemplo:**

*Digite a temperatura em Celsius: 30.0*

*Equivalente em Fahrenheit: 86.0*

**Deseja repetir (s/n)? s**

*Digite a temperatura em Celsius: 21.0*

*Equivalente em Fahrenheit: 69.8*

**Deseja repetir (s/n)? s**

*Digite a temperatura em Celsius: -10.5*

*Equivalente em Fahrenheit: 13.1*

**Deseja repetir (s/n)? n**

**Restrições e convenções para nomes**

Nomes de variáveis:

* Não pode começar com dígito: use uma letra ou \_
* Não usar acentos ou til
* Não pode ter espaço em branco
* Sugestão: use nomes que tenham um significado



Convenções

Camel Case: lastName

* pacotes;
* atributos;
* métodos
* variáveis e parâmetros

Pascal Case: ProductService

* classes

**Operadores bitwise**







**Funções interessantes para String**

Checklist

* **Formatar**: toLowerCase(),toUpperCase(),trim()
* **Recortar**: substring(inicio),substring(inicio, fim)
* **Substituir**: Replace(char, char), Replace(string, string)
* **Buscar**: IndexOf, LastIndexOf
* str.Split(“ “)

**Operação split**

*String s = “potato apple lemon”;*

*String [] vect = s.split(“ “);*

*String word1 = vect[0];*

*String word2 = vect[1];*

*String word3 = vect[2];*



**Funções**

Representam um processamento que possui um significado

* Math.sqrt(double)
* System.out.println(string)

Principais vantagens: modularização, delegação e reaproveitamento

Dados de entrada e saída

* Funções podem receber dados de entrada (parâmetros ou argumentos)
* Funções podem ou não retornar uma saída

Em orientação a objetos, funções em classes recebem o nome de “métodos”

**Problema exemplo:**

Fazer um programa para ler três números inteiros e mostrar na tela o maior deles.

Exemplo:

*Enter three numbers:*

**5**

**8**

**3**

*Higher = 8*

Vamos criar uma função logo após a função principal main.

Usaremos public static, public para publico (ficar disponível para outras classes) e static para que essa função possa ser chamada independente de criar um objeto.

Programa criado:

package problema01Aula61;

import java.util.Scanner;

public class Application {

public static void main(String[] args) {

Scanner keyboard = new Scanner(System.***in***);

System.***out***.println("Enter three numbers: ");

int a = keyboard.nextInt();

int b = keyboard.nextInt();

int c = keyboard.nextInt();

int higher = *max*(a, b, c);

*showResult*(higher);

keyboard.close();

}

public static int max(int x, int y, int z) {

int aux;

if (x > y && x > z) {

aux = x;

} else if (y > z) {

aux = y;

} else {

aux = z;

}

return aux;

}

public static void showResult(int value) {

System.***out***.println("Higher = " + value);

}

}

**Resolvendo um problema sem orientação a objetos**

Fazer um programa para ler as medida dos lados de dois triângulos X e Y (suponha medidas válidas). Em seguida, mostrar o valor das áreas dos dois triângulos e dizer qual dos dois triângulos possui a maior área.

A fórmula para calcular a área de um triângulo a partir das medidas de seus lados a, b e c é a seguinte (fórmula de Heron):



Exemplo:

Enter the measures of triangule X:

3.00

4.00

5.00

Enter the measures of triangle Y:

7.50

4.50

4.02

Triangle X area: 6.0000

Triangle Y area: 7.5638

Larger area: Y

**Criando uma classe com três atributos para representar melhor o triângulo**

Discussão

Triângulo é uma entidade com três atributos: a, b, c.

Estamos usando três variáveis distintas para representar cada triângulo:

double aX, bX, cX, aY, bY, cY;

Para melhorar isso, vamos usar uma CLASSE para representar um triângulo.

Memória:



**Classe**

É um tipo estrutura que pode conter (membros):

* Atributos (dados / campos)
* Métodos (funções /operações)

A classe também pode prover muitos outros recursos, tais como:

* Construtores
* Sobrecarga
* Encapsulamento
* Herança
* Polimorfismo

Exemplos:

* Entidades: produto, Cliente, Triangulo
* Serviços: ProdutoService, ClienteService, EmailService, StorageService
* Controladores: ProdutoController, ClienteController
* Utilitários: Calculadora, Compactador
* Outros (views, repositórios, gerenciadores, etc.)

package entities;

public class Triangle **{**

public double a;

public double b;

public double c;

**}**

Triangle x, y;

x = new Triangle();

y = new Triangle();





**Criando um método para obtermos os benefícios de reaproveitamento e delegação**

Discussão

Com o uso de CLASSE, agora nós temos uma variável composta do tipo “Triangle” para representar cada triângulo:

Triangle x, y;

x = new Triangle();

y = new Triangle();

Agora vamos melhorar nossa CLASSe acrescentando nela um MÉTODO para calcular a área.

**Memória:**





**Projeto de classe (UML)**



Benefícios de se calcular a área de um triângulo por meio de um **MÉTODO** dentro da **CLASSE**:

1. **Reaproveitamento de código:** nós eliminamos o código repetido (cálculo das áreas dos triângulos x e y) no programa principal.
2. **Delegação de responsabilidades:** quem deve ser responsável por saber como calcular a área não deve estar em outro lugar.

**Resolvendo um segundo problema exemplo.**

Fazer um programa para ler os dados de um produto em estoque (nome, preço e quantidade no estoque). Em seguida:

Mostrar os dados do produto (nome, preço, quantidade no estoque, valor total no estoque)

Realizar uma entrada no estoque e mostrar novamente os dados do produto.

Realizar uma saída no estoque e mostrar novamente os dados do produto.

Para resolver este problema, você deve criar uma CLASSE conforme projeto ao lado:



Exemplo:

Enter product data:

Name: TV

Price: 900.00

Quantity in stock: 10

Product data: TV, $ 900.00, 10 units, Total: $ 9000.00

Enter the number of products to be added in stock: 5

Update data: TV, $ 900.00, 15 units, Total: $ 13500.00

Enter the number of products to be removed from stock: 3

Updated data: TV, $ 900.00, 12 units, Total: $ 10800.00

**Object e toString**

**Discussão**

Toda classe em Java é uma subclasse da classe Object.

Object possui os seguintes métodos:

* getClass - retorna o tipo do objeto;
* equals - compara se o objeto é igual a outro;
* hashCode - retorna um código hash do objeto;
* toString - converte o objeto para string