
▼ FUNDAMENTOS - COMANDOS

▼ Sintaxe da Linguagem

Java Development Kit - JDK

É o pacote oficial que você precisa instalar para programar em Java.

Ele contém tudo que é necessário para escrever, compilar, executar e depurar programas Java.

O que o JDK contém?

- JRE (Java Runtime Environment)
 - Ambiente que permite executar programas Java.
 - Contém a JVM (Java Virtual Machine) + bibliotecas básicas.
 -  Se você só quiser rodar programas Java (não criar), basta o JRE.
- JVM (Java Virtual Machine)
 - A "máquina virtual" que executa o bytecode Java (o .class).
 - Torna o Java portável: "escreva uma vez, rode em qualquer lugar".
- Compilador (javac)
 - Transforma arquivos .java em arquivos .class (bytecode).
- Ferramentas de desenvolvimento
 - javac → Compilador | .java → .class
 - java → Executor
 - javadoc → Gera documentação
 - jar → Compacta bibliotecas/arquivos em .jar
 - jdb → Debugger

Estrutura Básica

Quando você instala o JDK, normalmente você vê pastas como:

- bin/ → onde ficam os executáveis (javac, java, javadoc...)
- lib/ → bibliotecas necessárias
- include/ → arquivos para integração com outras linguagens
- jmods/ (em versões recentes) → módulos do Java

Versões do JDK

- O Java é atualizado a cada 6 meses. Versões mais estáveis:

- Java 8
 - Java 11
 - Java 17
 - Java 21
- SE (Standard Edition) - Edição padrão
 - Aplicação, janelas, ambientes
 - EE (Enterprise Edition) - Edição para empresas
 - Acesso remoto, bases de dados gigantes, etc
 - ME (Micro Edition) - Edição pequena
 - Aplicações para smartphones, coisas pessoais, etc

Estrutura do Java

```
JDK (Java Development Kit)
|
|--- JRE (Java Runtime Environment)
|   |
|   |--- JVM (Java Virtual Machine)
|   |--- Bibliotecas básicas (API Java)
|
|--- Ferramentas de desenvolvimento
    |--- javac (compilador)
    |--- java (executor)
    |--- javadoc (documentação)
    |--- jar (empacotamento)
```

Fluxo de um programa Java

```
[ Você escreve ]
Programa.java (código fonte em Java)
|
|--- ▼ javac (compilador)
Programa.class (bytecode)
|
|--- ▼ java (executor)
JVM interpreta o bytecode
|
|--- ▼
Programa rodando no seu computador
```

A estrutura de um código em Java

- public class é a declaração do "nome do programa". O arquivo que contém o código Java deve ser salvo com o mesmo nome que aparece após a declaração public class e mais a extensão .java (o exemplo abaixo deveria ser salvo como NomeDoPrograma.java).
- As chaves indicam o início ou o fechamentos do blocos.
- Todo comando é terminado por um ponto e vírgula;
- O método main deve aparecer em todos os códigos Java. Quando um programa Java é executado, o interpretador da JVM executa os comandos que estiverem dentro do bloco indicado pelo método "static public void main(String)".

```
public class NomeDoPrograma {  
  
    static public void main(String[] args) {  
        // aqui virão os comandos  
    }  
  
}
```

Explicação do método main:

- static: um modificador utilizado pelo compilador para identificar métodos que podem ser executados apenas no contexto da classe AloMundo, sem a necessidade que um objeto dessa classe seja instanciada.
- public: o método main pode ser executado por qualquer processo ativo no sistema operacional, incluindo o interpretador Java.
- void: indica o tipo do valor (int, char, etc.) a ser retornado pelo método main. Quando um tipo de retorno é declarado como void, significa que o método não retorna nenhum valor. O método main sempre deverá ser declarado static public void. Caso contrário o programa não poderá ser executado (Exception in thread "main" java.lang.NoSuchMethodError: main).
- String[] args: um array de objetos do tipo String, que serve para armazenar a lista de argumentos digitados na linha de comando após o nome da classe a ser executada:

Código Fonte

Como qualquer outra linguagem de programação, Java é usada para criar aplicações de computador. O texto que contém os comandos a serem executados pela JVM é chamado de código-fonte, ou simplesmente fonte.

Comentários no Java

Comentários em Java são uma maneira de adicionar notas e explicações ao código, tornando-o mais legível e compreensível para outros desenvolvedores (ou para você mesmo no futuro).

Boas Práticas

- Seja Claro e Conciso: Evite comentários desnecessários. Comentários devem adicionar valor e esclarecer o que o código faz.
- Atualize os Comentários: Mantenha os comentários atualizados com o código. Comentários desatualizados podem levar a mal-entendidos.
- Use Comentários para Explicar “Por Quê”: Em vez de simplesmente descrever o que o código faz (o que o código em si já deve mostrar), use comentários para explicar por que algo é feito de uma certa maneira.
- Evite Comentários Excessivos: Comentários em excesso podem poluir o código. Apenas adicione comentários onde realmente são necessários.

Comentário de linha única

Usado para anotações curtas.

Começa com // e vai até o final da linha.

```
// Este é um comentário
int idade = 25; // Pode ser usado no fim da linha também
```

Comentário em Bloco

Usado para explicações mais longas.

Começa com /* e termina com */.

```
/* Este é um comentário
que pode ocupar
várias linhas */
```

Comentário de documentação

Usado para gerar documentação automática com a ferramenta javadoc.

Começa com /** e termina com */.

Permite tags especiais como @param, @return, @author.

```
/**
 * Calcula a soma de dois números inteiros.
 *
 * @param a primeiro número
 * @param b segundo número
 * @return a soma de a e b
 */
```

```
public int soma(int a, int b) {  
    return a + b;  
}
```

Identificadores e Palavras Reservadas

Identificadores Válidos

Em Java, um identificador é uma seqüência de símbolos UNICODE (64K símbolos) que começa com uma letra, um símbolo subscrito `_`, ou o caractere `$`.

Os demais símbolos de um identificador podem conter também números. Identificadores são case-sensitive e não tem um tamanho máximo estabelecido.

Apesar da tabela UNICODE ser bastante extensa, um bom hábito de programação é utilizar somente letras do alfabeto (a-Z) e números para nomear identificadores.

Exemplo de identificadores válidos em Java:

- `data`
- `_data`
- `$data`
- `data_do_mês`
- `data1`
- `uma_variável_pode_SER_bastante_extensa_e_conter_Numeros234876238476`
- `data_public_class_NoteQueEssaIdentificadorContémPalavrasReservadas`

Palavras Reservadas

Apesar desta "liberdade" de opções para nomes de identificadores, algumas palavras não são permitidas. Tais palavras são ditas palavras reservadas, e representam o conjunto de comandos que forma a sintaxe da linguagem Java.

Declaração de Variáveis

Uma variável é sempre declarada seguindo do seguinte esquema:

tipo + <espaço> + identificador + ;

ou

tipo + <espaço> + identificador + = + valor + ; onde:

- **tipo** é um tipo primitivo de dados ou o nome de uma classe ou interface
- **identificador** é o nome da variável
- **valor** é o valor atribuído à variável. Caso você declare uma variável e não atribua nenhum valor, ela não poderá ser utilizada em um código Java – a tentativa de utilizar uma variável não inicializada em Java gerará um erro de compilação.

```

public class AloMundo
{
    static public void main(String[] args)
    {
        boolean obrigatorio;
        int semestre = 2;
        String mensagem = "Alo mundo.";

        System.out.println(mensagem);
    }
}

```

Convenções de Codificação

Classes – as classes devem ser designadas por nomes, começando por uma letra maiúscula e depois minúsculas. Cada nova palavra que formar o nome da classe deve ser capitalizada. Ex: class Calculadora, class CalculadoraCientifica, ...

Interfaces – igual às classes. Ex: interface Calculo, interface EquacaoLogaritmica, ...

Métodos – métodos devem nomeados por verbos, seguindo o mesmo formato de capitalização das classes. Entretanto, um método sempre deve começar com letra minúscula. Ex: public void calcular(int numero), public void extrairRaiz(int numero), ...

Constantes – constantes devem ter todas as suas letras em maiúsculo, com o símbolo de subscrito para separar as palavras. Ex: final int ANO = 2002, final boolean VERDADE = true, ...

Variáveis – tal qual os métodos, as variáveis devem começar com uma letra minúscula e depois alternar a cada palavra. Procure usar nomes significativos para variáveis. Evite declarar variáveis usando apenas uma letra.

▼ Tipos Primitivos e Operadores

Tipos Primitivos

| Tipo Primitivo | Tipo Primitivo | Tamanho | Intervalo | Exemplo |
|------------------------------|----------------|---------|--------------------------------|--|
| Numéricos Inteiros | byte | 8 bits | -128 e 127 | byte b = 18; |
| Numéricos Inteiros | short | 16 bits | -32.768 e 32.767 | short s = 300; |
| Numéricos Inteiros | int | 32 bits | -2.147.483.648 e 2.147.483.647 | int i = 50000; Pa |
| Numéricos Inteiros | long | 64 bits | números muito grandes | long l = 100000L; Pr |
| Numéricos de Ponto Flutuante | float | 32 bits | Precisão simples ~7 dígitos | float f = 3.14f; |
| Numéricos de Ponto Flutuante | double | 64 bits | Alta precisão ~15 dígitos | double d = 3.1415; Pa |

| Tipo Primitivo | Tipo Primitivo | Tamanho | Intervalo | Exemplo |
|----------------|----------------|----------|--------------------------------------|----------------------|
| Textuais | char | 16 bits | Armazena um único caractere | char c = 'A'; |
| Textuais | String | Variável | Armazena uma sequência de caracteres | Pa |
| Lógico | boolean | 1 bit | true ou false | boolean flag = true; |

String não é um tipo primitivo

Demonstração - Tipos Primitivos

comando

```
byte b = 120;
short s = 32000;
int i = 2_000_000;
long l = 9_000_000_000L;
float f = 3.14f;
double d = 3.14159265359;
char c1 = 'A';
char c2 = 65;
boolean ativo = true;
boolean maiorDeIdade = i > 18;
String nome = "Wictor";
String sobrenome = new String("Benevenutti");
```

```
System.out.println("BYTE: " + b);
System.out.println("SHORT: " + s);
System.out.println("INT: " + i);
System.out.println("LONG: " + l);
System.out.println("FLOAT: " + f);
System.out.println("DOUBLE: " + d);
System.out.println("CHAR (direto): " + c1);
System.out.println("CHAR (unicode 65): " + c2);
System.out.println("BOOLEAN (ativo): " + ativo);
System.out.println("BOOLEAN (maior de idade?): " + maiorDeIdade);
System.out.println("NOME: " + nome);
System.out.println("SOBRENOME: " + sobrenome);
```

saída

```
BYTE: 120
SHORT: 32000
INT: 2000000
LONG: 9000000000
FLOAT: 3.14
DOUBLE: 3.14159265359
CHAR (direto): A
CHAR (unicode 65): A
BOOLEAN (ativo): true
```

```
BOOLEAN (maior de idade?): true
```

```
NOME: Wictor
```

```
SOBRENOME: Benevenutti
```

Constantes

Variáveis primitivas podem ser marcadas como constantes com `final` na declaração da variável.

```
final double PI = 3.14159;
```

Overflow e Underflow

Se passar do limite do tipo, o valor “dá a volta”.

```
byte b = 127;  
b++; // agora b = -128 (overflow!)
```

Casting

Widening Casting (Conversão Implícita)

- Quando você converte de um tipo menor para um maior (não há risco de perda de informação).
- Feito automaticamente pelo Java.
- Ordem dos tipos numéricos (do menor para o maior):

`byte` → `short` → `int` → `long` → `float` → `double`

```
int num = 100;  
double valor = num; // implícito (int -> double)  
System.out.println(valor); // 100.0
```

Narrowing Casting (Conversão Explícita)

- Quando você converte de um tipo maior para um menor.
- Pode haver perda de dados ou mudança no valor.
- Precisa indicar manualmente com `(tipo)`.

```
double valor = 9.78;  
int num = (int) valor; // explícito (double -> int)  
System.out.println(num); // 9
```

📌 O decimal foi perdido, só ficou a parte inteira.

Problema de Overflow

Se você converter para um tipo menor do que o valor suporta, ele “dá a volta”.

```
int grande = 130;
byte pequeno = (byte) grande; // 130 não cabe em byte (-128 a 127)
System.out.println(pequeno); // -126 (overflow!)
```

Casting entre primitivos e objetos (Wrappers)

Às vezes você precisa converter Strings para números (parsing).

```
String texto = "123";
int numero = Integer.parseInt(texto); // String -> int
System.out.println(numero + 1); // 124
```

Wrapper Classes

São classes que empacotam (wrap) um valor primitivo dentro de um objeto.

Cada tipo primitivo tem seu Wrapper:

| Primitivo | Wrapper (Classe) |
|-----------|------------------|
| byte | Byte |
| short | Short |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |
| char | Character |
| boolean | Boolean |

Coleções (List, Map, Set) só trabalham com objetos, não com primitivos.

Métodos utilitários → Wrappers têm métodos úteis (parse, valueOf, compareTo, etc.).

Constantes úteis (ex.: Integer.MAX_VALUE, Double.MIN_VALUE).

Operadores Unários

Operador - (inverte o sinal)

Operador ++ (Incremento) – incremento de 1 unidade | a++ | a = a + 1

Operador -- (Decremento) - decremento de 1 unidade | a-- | a = a - 1

📌 Pré-incremento/decremento → altera antes de usar.

💡 Pós-incremento/decremento → altera depois de usar.

```
int a = 5;

System.out.println(++a); // pré-incremento: a vira 6 e imprime 6
System.out.println(a++); // pós-incremento: imprime 6, depois vira 7

int b = 5;
System.out.println(--b); // pré-decremento: b vira 4 e imprime 4
System.out.println(b--); // pós-decremento: imprime 4, depois vira 3
```

Demonstração - Operadores Unários

comando

```
int numeroinc = 5;
int numerodec = 10;

System.out.println("Número original incremento: " + numeroinc);
System.out.println("Pré-incremento (++numero): " + (++numeroinc));
System.out.println("Valor após pré-incremento: " + numeroinc);
System.out.println("Pós-incremento (numero++): " + (numeroinc++));
System.out.println("Valor final após pós-incremento: " + numeroinc);

System.out.println("\nNúmero original decremeno: " + numerodec);
System.out.println("Pré-decremeno (--numero): " + (--numerodec));
System.out.println("Valor após pré-decremeno: " + numerodec);
System.out.println("Pós-decremeno (numero--): " + (numerodec--));
System.out.println("Valor final após pós-decremeno: " + numerodec);
```

saída

```
Número original incremento: 5
Pré-incremento (++numero): 6
Valor após pré-incremento: 6
Pós-incremento (numero++): 6
Valor final após pós-incremento: 7
```

```
Número original decremeno: 10
Pré-decremeno (--numero): 9
Valor após pré-decremeno: 9
Pós-decremeno (numero--): 9
Valor final após pós-decremeno: 8
```

Operadores Aritméticos

| Operação | Operador | Descrição |
|----------------|----------|--|
| Adição | + | Realiza a soma entre operandos |
| Subtração | - | Realiza a subtração entre operandos e adiciona o sinal de negativo ao número |
| Multiplicação | * | Realiza a multiplicação entre operandos |
| Divisão | / | Realiza a divisão entre operandos |
| Módulo (resto) | % | Retorna o resto da divisão entre operandos |

Demonstração - Operadores Aritméticos

comando

```
int a = 15;  
int b = 4;  
  
int soma = a + b;  
int subtracao = a - b;  
int multiplicacao = a * b;  
int divisaoInteira = a / b;  
double divisaoDecimal = (double) a / b;  
int resto = a % b;  
  
System.out.println("Adição (a + b): " + soma);  
System.out.println("Subtração (a - b): " + subtracao);  
System.out.println("Multiplicação (a * b): " + multiplicacao);  
System.out.println("Divisão inteira (a / b): " + divisaoInteira);  
System.out.println("Divisão com decimais ((double)a / b): " + divisaoDecimal);  
System.out.println("Módulo (a % b): " + resto);
```

saída

```
Adição (a + b): 19  
Subtração (a - b): 11  
Multiplicação (a * b): 60  
Divisão inteira (a / b): 3  
Divisão com decimais ((double)a / b): 3.75  
Módulo (a % b): 3
```

Operadores de Atribuição

| Operação | Operador | Exemplo | Equivalente |
|------------------------|----------|---------|-------------|
| Atribuição Simples | = | a = 1 | a = 1 |
| Somar e Atribuir | += | a += b | a = a + b |
| Subtrair e Atribuir | -= | a -= b | a = a - b |
| Multiplicar e Atribuir | *= | a *= b | a = a * b |

| Operação | Operador | Exemplo | Equivalente |
|--------------------|-----------------|---------------------|------------------------|
| Dividir e Atribuir | <code>/=</code> | <code>a /= b</code> | <code>a = a / b</code> |
| Resto e Atribuir | <code>%=</code> | <code>a %= b</code> | <code>a = a % b</code> |

Demonstração - Operadores de Atribuição

comando

```
int a = 10;
```

```
System.out.println("Valor inicial de a: " + a);
System.out.println("Após a += 5: " + (a += 5));
System.out.println("Após a -= 3: " + (a -= 3));
System.out.println("Após a *= 2: " + (a *= 2));
System.out.println("Após a /= 4: " + (a /= 4));
System.out.println("Após a %= 3: " + (a %= 3));
```

saída

```
Valor inicial de a: 10
Após a += 5: 15
Após a -= 3: 12
Após a *= 2: 24
Após a /= 4: 6
Após a %= 3: 0
```

Operadores Relacionais

- Operadores relacionais retornam boolean. true ou false.
- Para números → usamos `==`, `!=`, `>`, `<`, `>=`, `<=`.
- Para String → usamos `equals()`.

| Operação | Operador | Descrição |
|------------------|----------------------------------|---|
| Maior que | <code>></code> | Verifica se um valor é maior que outro |
| Menor que | <code><</code> | Verifica se um valor é menor que outro |
| Maior ou igual a | <code>>=</code> | Verifica se um valor é maior ou igual a outro |
| Menor ou igual a | <code><=</code> | Verifica se um valor é menor ou igual a outro |
| Igual a | <code>==</code> | Verifica se um valor é igual a outro |
| Diferente de | <code>!=</code> | Verifica se um valor é diferente de outro |
| Strings Iguais | <code>.equals()</code> | Verifica se uma string é igual a outra |
| Strings Iguais | <code>.equalsIgnoreCase()</code> | Ignora maiúsculas e minúsculas na comparação de strings |

Demonstração - Operadores Relacionais

comando

```
int x = 10;
```

```

int y = 20;
boolean bool1 = true;

System.out.println("Igual a (x == y) -> " + (x == y));
System.out.println("Diferente de (x != y) -> " + (x != y));
System.out.println("Maior que (x > y) -> " + (x > y));
System.out.println("Menor que (x < y) -> " + (x < y));
System.out.println("Maior ou igual a (x >= 10) -> " + (x >= 10));
System.out.println("Menor ou igual a (y <= 20) -> " + (y <= 20));
System.out.println("Boolean igual a (bool1 == false) -> " + (bool1 == false));
System.out.println("Boolean diferente de (bool1 != false) -> " + (bool1 != false));

```

saída

```

Igual a (x == y) -> false
Diferente de (x != y) -> true
Maior que (x > y) -> false
Menor que (x < y) -> true
Maior ou igual a (x >= 10) -> true
Menor ou igual a (y <= 20) -> true
Boolean igual a (bool1 == false) -> false
Boolean diferente de (bool1 != false) -> true

```

Operadores Lógicos

- Usados para condições complexas, sempre retornam boolean
- curto-circuito (`&&` e `||` possuem curto-circuito)
 - `&&` → se a primeira expressão for `false`, a segunda não é avaliada
 - `||` → se a primeira expressão for `true`, a segunda não é avaliada
- Você pode combinar vários operadores com parênteses para controlar a ordem.

Operador `&&` (and) – retorna `true` se todas as condições forem verdadeiras, caso contrário retorna `false`.

Operador `||` (or) – retorna `true` se uma das condições for verdadeira, caso contrário retorna `false`.

Operador `^` (or exclusivo) – retorna `true` se apenas uma das condições for verdadeira, caso contrário retorna `false`.

Operador `!` (not) – inverte o resultado: se o resultado da expressão for `true`, o operador retorna `false`, e vice-versa.

Demonstração - Operadores Lógicos

```

comando
int a = 10;
int b = 20;
int c = 30;

System.out.println("AND: (a < b) && (b < c) -> " + ((a < b) && (b < c)));
System.out.println("OR: (a > b) || (b < c) -> " + ((a > b) || (b < c)));
System.out.println("NOT: !(a < b) -> " + (!(a < b)));

```

saída

```

AND: (a < b) && (b < c) -> true
OR: (a > b) || (b < c) -> true
NOT: !(a < b) -> false

```

▼ Classes Utils e Lang

Classe System

Faz parte do pacote `java.lang` e inicia automaticamente.

Contém métodos e objetos estáticos usados para interagir com o sistema e com o console.

Métodos Comuns da Classe

| Método | Descrição | Exemplo |
|---|---|--|
| <code>System.exit(int status)</code> | Encerra o programa. 0 = normal, 1 ou outros = erro | <code>System.exit(0);</code> |
| <code>System.currentTimeMillis()</code> | Retorna tempo atual em milissegundos desde 1/1/1970 | <code>long t = System.currentTimeMillis();</code> |
| <code>System.gc()</code> | Sugere execução do Garbage Collector | <code>System.gc();</code> |
| <code>System.arraycopy()</code> | Copia elementos de um array para outro | <code>System.arraycopy(origem, origem, destino, destino, quantidade);</code> |
| <code>System.getProperty(String key)</code> | Pega propriedade do sistema | <code>System.getProperty("os.name");</code> |

Objeto System.in

Para ler dados do teclado (normalmente usado com `Scanner`).

Objeto System.err

Para imprimir mensagens de erro no console.

Objeto System.out

- Ele é um objeto estático da classe `System`.
- Seu tipo é `PrintStream`.
- É responsável por enviar saída de texto para o console (a tela do terminal).
- Ou seja, sempre que você faz um `System.out.println(...)`, está chamando métodos da classe `PrintStream`.

Principais Métodos do System.out

| Método | O que faz | Exemplo |
|-------------------------|--|--|
| print(x) | Imprime o valor sem quebrar linha | System.out.print("Olá"); |
| println(x) | Imprime o valor e quebra a linha | System.out.println("Mundo!"); |
| printf(format, args...) | Imprime com formatação , estilo C | System.out.printf("Idade: %d anos", 25); |
| format(format, args...) | Igual ao printf, mas mais semântico | System.out.format("Nome: %s", "Ana"); |
| append(c) | Adiciona texto ao final da saída | System.out.append("Texto extra"); |
| flush() | Força a saída de dados que estavam em buffer | System.out.flush(); |

- **print()** e **println()**

- colocam a saída em buffer antes de mostrar no console.
- Aceitam todos os tipos primitivos e objetos (via `toString()`).

```
Date agora = new Date();
System.out.println(agora.toString());
```

- Podem ser usados com concatenação (+).
- `println()` vazio apenas pula linha.

- **flush()**

- No `System.out`, os dados são primeiro colocados no buffer e só depois aparecem no console.
- Força a saída de dados que estavam em buffer (memória temporária para armazenar dados antes de processar).
- Quando o buffer é esvaziado?
 - O programa termina.
 - Você usa `println` (que geralmente força a liberação).
 - Você chama manualmente `System.out.flush()`.
 - O buffer fica cheio.

```
System.out.print("Olá ");
System.out.print("mundo");
// até aqui pode ser que ainda esteja no buffer
System.out.flush(); // força a saída imediata
```

- **printf()**

| Especificador | Tipo / Uso | Variações comuns | Exemplo |
|---------------|--|---|------------|
| %d | Inteiro decimal | %5d → largura mínima 5 %05d → zeros à esquerda | System.out |
| %f | Ponto flutuante (decimal) | %.2f → 2 casas decimais %10.3f → largura 10, 3 casas | System.out |
| %e ou %E | Notação científica | %.3e → 3 casas decimais | System.out |
| %g ou %G | Usa %f ou %e automaticamente (o mais compacto) | %.4g → até 4 dígitos significativos | System.out |

| Especificador | Tipo / Uso | Variações comuns | Exemplo |
|---------------|-----------------------|--|------------|
| %s | String | %10s → largura 10, alinhado à direita %-10s → alinhado à esquerda | System.out |
| %c | Caractere | — | System.out |
| %b | Booleano | — | System.out |
| %n | Nova linha (portável) | — | System.out |
| %% | Imprimir % | — | System.out |

```

int num = 42;
double pi = 3.1415926535;
String nome = "Java";

System.out.printf("Inteiro normal: %d%n", num); //Inteiro normal: 42
System.out.printf("Inteiro com zeros: %05d%n", num); //Inteiro com zeros: 00042
System.out.printf("Decimal com 2 casas: %.2f%n", pi); //Decimal com 2 casas: 3.14
System.out.printf("Decimal notação científica: %e%n", pi); //Decimal notação científica: 3.1415926535
System.out.printf("Texto alinhado à direita: |%10s|%n", nome); //Texto alinhado à direita: |Java|n
System.out.printf("Texto alinhado à esquerda: |%-10s|%n", nome); //Texto alinhado à esquerda: |Java|n
System.out.printf("Booleano: %b%n", true); //Booleano: true
System.out.printf("Caractere: %c%n", 'J'); //Caractere: J
System.out.printf("Imprimindo %: 100%%%n"); //Imprimindo %: 100%

```

Classe Scanner

É necessário importar com "import java.util.Scanner;".

Cria um objeto que permite capturar dados de forma fácil. Serve para ler entradas de teclado, arquivos e strings.

Criando um Objeto Scanner para teclado

System.in indica que a entrada será do teclado. sc é o nome do objeto que será utilizado depois para capturar dados do teclado.

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

Principais Métodos

| Método | Descrição | Exemplo |
|-----------------------------------|--------------|-------------------------------|
| Lê a próxima palavra (até espaço) | next() | String palavra = sc.next(); |
| Lê uma linha inteira | nextLine() | String linha = sc.nextLine(); |
| Lê um inteiro | nextInt() | int n = sc.nextInt(); |
| Lê um número decimal | nextDouble() | double d = sc.nextDouble(); |
| Lê um float | nextFloat() | float f = sc.nextFloat(); |
| Lê um long | nextLong() | long l = sc.nextLong(); |

| Método | Descrição | Exemplo |
|-------------------------------|---------------|-------------------------------|
| Lê um boolean (true ou false) | nextBoolean() | boolean b = sc.nextBoolean(); |

```

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Digite seu nome: ");
        String nome = sc.nextLine();

        System.out.print("Digite sua idade: ");
        int idade = sc.nextInt();

        sc.close(); // fecha o Scanner
    }
}

```

Dicas

- Sempre feche o Scanner com sc.close() para liberar recursos.
- Tome cuidado ao misturar nextLine() com nextInt()/nextDouble(), porque os números não consomem a quebra de linha. Solução: adicionar um sc.nextLine() extra depois de ler números.

```

int idade = sc.nextInt();
sc.nextLine(); // consome o "\n" deixado pelo nextInt()
String nome = sc.nextLine();

```

- Pode ler arquivos usando: Scanner sc = new Scanner(new File("arquivo.txt"));

Classe Math

Faz parte do pacote java.lang e inicia automaticamente.

Constantes

| Operação | Exemplo | Resultado |
|-----------------|---------|-------------------|
| Número PI | Math.PI | 3.141592653589793 |
| Número de Euler | Math.E | 2.718281828459045 |

Operações Básicas

| Operação | Exemplo | Resultado |
|----------|-----------------|-----------|
| Absoluto | Math.abs(-5) | 5 |
| Máximo | Math.max(10,20) | 20 |

| Operação | Exemplo | Resultado |
|---------------|-----------------|-----------|
| Mínimo | Math.min(10,20) | 10 |
| Raiz Quadrada | Math.sqrt(25) | 5.0 |
| Raiz Cúbica | Math.cbrt(27) | 3.0 |
| Potência | Math.pow(2,3) | 8.0 |

Arredondamento

| Operação | Exemplo | Resultado |
|----------------------|-------------------------------|-----------|
| Arredonda para Cima | Math.ceil(5.2) | 6 |
| Arredonda para Baixo | Math.floor(5.9) | 5 |
| Arredonda | Math.round(5.7) | 6 |
| Arredonda 1 casa | Math.round(5.7 * 10) / 10 | 5.70 |
| Arredonda 2 casas | Math.round(5.7 * 100) / 100 | 5.70 |
| Arredonda 3 casas | Math.round(5.7 * 1000) / 1000 | 5.70 |

Trigonometria (em radianos)

| Operação | Exemplo |
|---------------------|-------------------|
| Seno | Math.sin(x) |
| Cosseno | Math.cos(x) |
| Tangente | Math.tan(x) |
| Graus para Radianos | Math.toRadians(x) |
| Radianos para Graus | Math.toDegrees(x) |

Exponenciais e Logaritmos

| Operação | Exemplo | Resultado |
|-----------------------------|------------------|-----------|
| Potência de Euler | Math.exp(1) | 2.718... |
| Logaritmo Natural na Base e | Math.log(Math.E) | 1 |
| Logaritmo na Base 10 | Math.log10(1000) | 3 |

Aleatório

| Operação | Exemplo | Resultado |
|-------------------|---|-----------------|
| Aleatório Padrão | Math.random() | Entre 0.0 e 1.0 |
| Aleatório 1 até x | (int)(Math.random() * 6) +1 | Entre 1 e 6 |
| Aleatório x até y | (int)(Math.random() * (50 - 20 + 1)) + 20 | Entre 20 e 50 |

Classe String

Faz parte do pacote `java.lang` e inicia automaticamente.

Fatiamento

| Descrição | Método | Exemplo | Resultado |
|----------------------------|-------------------------------|--------------------------------|-----------|
| Caractere na Posição | charAt(int index) | "Java".charAt(2) | "V" |
| Parte da String | substring(int start, int end) | "Java".substring(1,3) | "AV" |
| Retorna a primeira palavra | split(" ") | "Java Linguagem".split(" ")[0] | "Java" |

Informações

| Descrição | Método | Exemplo | Resultado |
|------------------------------|--------------------------|---------------------------------|-----------|
| Tamanho da String | length() | "Java".length() | 4 |
| Contém | contains(CharSequence s) | "Java".contains("av") | true |
| Posição 1º ocorrência | indexOf(String s) | "Java".indexOf("v") | 2 |
| Posição último ocorrência | lastIndexOf(String s) | "Java".lastIndexOf("a") | 3 |
| Verifica se começa com algo | startsWith() | "Java é top".startsWith("Java") | true |
| Verifica se termina com algo | endsWith() | "Java é top".endsWith("top") | true |

Alterações na String

| Descrição | Método | Exemplo | Resultado |
|--------------------------------|---|-------------------------|--------------|
| Maiúsculas | toUpperCase() | "Java".toUpperCase() | "JAVA" |
| Minúsculas | toLowerCase() | "Java".toLowerCase() | "java" |
| Remove Espaços no Início e Fim | trim() | " Java ".trim() | "Java" |
| Substitui | replace(CharSequence old, CharSequence new) | "Java".replace("a","o") | "Jovo" |
| Concatena Duas Strings | concat(String s) | "Java".concat(" é top") | "Java é top" |

Comparação de Strings

| Descrição | Método | Exemplo | Resultado |
|---|----------------------------|---------------------------------|-----------|
| Compara Strings | equals(String s) | "Java".equals("java") | false |
| Compara Strings Ignorando Maiúsculas e Minúsculas | equalsIgnoreCase(String s) | "Java".equalsIgnoreCase("java") | true |

Conversão

| Descrição | Método | Exemplo | Resultado |
|--------------------------|----------------------|----------------------------|-----------|
| Outros tipos para String | String.valueOf() | String.valueOf(123) | "123" |
| String para int | Integer.parseInt() | Integer.parseInt("123") | 123 |
| String para double | Double.parseDouble() | Double.parseDouble("1.23") | 1.23 |

Classe Random

É necessário importar com "import java.util.Random;".

Usado para gerar números aleatório de forma mais flexível.

Permite gerar números aleatórios inteiros, decimais e booleanos.

Criando um Objeto Random

objetorandom é o nome do objeto que será utilizado depois.

```
Random objetorandom = new Random();
```

Principais Métodos

| Descrição | Método | Retorno | Exemplo |
|--|--------------------|---------|----------------------------|
| Número inteiro aleatório (positivo ou negativo) | nextInt() | int | random.nextInt() |
| Número inteiro entre 0 (inclusive) e bound (exclusive) | nextInt(int bound) | int | random.nextInt(10) → 0 a 9 |

| Descrição | Método | Retorno | Exemplo |
|--------------------------------|---------------|---------|----------------------|
| Número decimal entre 0.0 e 1.0 | nextDouble() | double | random.nextDouble() |
| Número decimal entre 0.0 e 1.0 | nextFloat() | float | random.nextFloat() |
| Número longo aleatório | nextLong() | long | random.nextLong() |
| true ou false aleatório | nextBoolean() | boolean | random.nextBoolean() |

```
int inteiro = random.nextInt(100); // 0 a 99
double decimal = random.nextDouble(); // 0.0 a 1.0
boolean bool = random.nextBoolean();
```

Gerando Números em Um Intervalo Específico

Para gerar de min a max inclusive:

```
int min = 25;
int max = 50;

int numero = random.nextInt(max - min + 1) + min;
```

▼ Condisional Simples e Composta

As estruturas condicionais em Java são usadas para tomar decisões no programa, ou seja, executar diferentes blocos de código dependendo de uma condição (true ou false).

Temos as seguintes opções:

- **Operador Ternário** → Útil para atribuição direta.
- **if** → Uma condição simples.
- **if...else** → Duas opções.
- **if...else if...else** → Várias condições.
- **switch** → Melhor que muitos ifs encadeados.

Operador Ternário

- Forma curta de escrever um if...else.
- Útil quando você precisa escolher entre dois valores rapidamente.

Estrutura

```
variavel = (condicao) ? valorSeTrue : valorSeFalse;
```

Exemplo

```
int idade = 20;  
String resultado = (idade >= 18) ? "Maior de idade" : "Menor de idade";  
  
System.out.println(resultado); // "Maior de idade"
```

if

Executa apenas se a condição for verdadeira.

Estrutura

```
if (condicao) {  
    // código executado se condicao for true  
}
```

Exemplo

```
int idade = 20;  
if (idade >= 18) {  
    System.out.println("Você é maior de idade.");  
}
```

if...else

Permite escolher entre dois caminhos: um se a condição for verdadeira, outro se for falsa.

Estrutura

```
if (condicao) {  
    // executa se true  
} else {  
    // executa se false  
}
```

Exemplo

```
int idade = 16;  
if (idade >= 18) {  
    System.out.println("Maior de idade.");  
} else {  
    System.out.println("Menor de idade.");  
}
```

if...else if...else

- Permite verificar múltiplas condições em sequência.
- Executa o bloco do primeiro if/else if que for verdadeiro.
- O else final é opcional, executa se nenhuma condição for verdadeira.

Estrutura

```
if (condicao1) {  
    // se condicao1 for true  
} else if (condicao2) {  
    // se condicao1 for false e condicao2 for true  
} else {  
    // se nenhuma condição for true  
}
```

Exemplo

```
int nota = 75;  
  
if (nota >= 90) {  
    System.out.println("Excelente");  
} else if (nota >= 60) {  
    System.out.println("Aprovado");  
} else {  
    System.out.println("Reprovado");  
}
```

switch

- Comparação de uma variável contra vários valores possíveis.
- Cada valor é um case.
- Use break para parar após encontrar o caso correto.
- default funciona como o else do if.

Estrutura

```
switch (variavel) {  
    case valor1:  
        // código  
        break;  
    case valor2:  
        // código  
        break;  
    default:}
```

```
// executa se nenhum case bater
}
```

Exemplo

```
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;
    default:
        System.out.println("Dia inválido");
}
```

▼ FUNDAMENTOS - EXERCÍCIOS VARIADOS

▼ Exercícios Tratando dados e fazendo contas

exercício 1 - deixando tudo pronto

Crie um programa que escreva "Olá, Mundo!" na tela

comando
String texto = "Olá, Mundo!";
System.out.print(texto);

Saída
Olá, Mundo!

exercício 2 - respondendo ao usuário

Faça um programa que leia o nome de uma pessoa e mostre a mensagem de boas-vindas.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o seu nome: ");
String nome = teclado.nextLine();
System.out.println("Seja Bem-Vindo " + nome);
```

saída

```
Digite o seu nome: Wictor
Seja Bem-Vindo Wictor
```

exercício 3 - somando dois números

Crie um programa que leia dois números e mostre a soma entre eles.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número: ");
float num1 = teclado.nextFloat();
System.out.print("Digite outro número: ");
float num2 = teclado.nextFloat();

float soma = num1 + num2;

System.out.printf("A soma entre os números %.2f e %.2f é igual a %.2f", num1, num2, soma);
```

saída

```
Digite um número: 2,7
Digite outro número: 4,1
A soma entre os números 2,70 e 4,10 é igual a 6,80
```

exercício 4 - antecessor e sucessor

Faça um programa que leia um número inteiro e mostre na tela o seu sucessor e seu antecessor.

```
comando
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número inteiro: ");
int num = teclado.nextInt();

int sucessor = num + 1;
int antecessor = num - 1;

System.out.println("Número digitado: " + num);
System.out.println("Antecessor: " + antecessor);
System.out.println("Sucessor: " + sucessor);
```

saída
Digite um número inteiro: 4
Número digitado: 4
Antecessor: 3
Sucessor: 5

exercício 5 - dobro, triplo, raiz quadrada

Crie um algoritmo que leia um número e mostre o seu dobro, triplo e raiz quadrada.

```
comando
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número: ");
float num = teclado.nextFloat();

float dobro = num * 2;
float triplo = num * 3;
float raiz = (float) Math.sqrt(num);

System.out.println("O dobro de " + num + " é " + dobro);
System.out.println("O triplo de " + num + " é " + triplo);
System.out.println("A raiz quadrada de " + num + " é " + raiz);
```

saída
Digite um número: 4
O dobro de 4.0 é 8.0

```
O triplo de 4.0 é 12.0
A raiz quadrada de 4.0 é 2.0
```

exercício 6 - média aritmética

Desenvolva um programa que leia as duas notas de um aluno, calcule e mostre a sua média.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Nota 1: ");
int nota1 = teclado.nextInt();
System.out.print("Nota 2: ");
int nota2 = teclado.nextInt();

int media = (nota1 + nota2) / 2;

System.out.print("Média das Notas: " + media);
```

saída

```
Nota 1: 7
Nota 2: 9
Média das Notas: 8
```

exercício 7 - conversor de medidas

Escreva um programa que leia um valor em metros e o exiba convertido em centímetros e milímetros.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Metros para Conversão: ");
float metros = teclado.nextFloat();

float centimetros = metros * 100;
float milimetros = metros * 1000;

System.out.println("Convertido para Centímetros: " + centimetros);
System.out.println("Convertido para Milímetros " + milimetros);
```

```
saída
Metros para Conversão: 4,2
Convertido para Centímetros: 419.99997
Convertido para Milímetros 4200.0
```

exercício 8 - tabuada

Faça um programa que leia um número inteiro qualquer e mostre na tela a sua tabuada.

comando

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

```
Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número inteiro: ");
int num = teclado.nextInt();
System.out.println("--- Tabuada do Número " + num + " ---");
System.out.println(num + " x 0 = " + num * 0);
System.out.println(num + " x 1 = " + (num * 1));
System.out.println(num + " x 2 = " + num * 2);
System.out.println(num + " x 3 = " + num * 3);
System.out.println(num + " x 4 = " + num * 4);
System.out.println(num + " x 5 = " + num * 5);
System.out.println(num + " x 6 = " + num * 6);
System.out.println(num + " x 7 = " + num * 7);
System.out.println(num + " x 8 = " + num * 8);
System.out.println(num + " x 9 = " + num * 9);
System.out.println(num + " x 10 = " + num * 10);
```

saída

```
Digite um número inteiro: 4
--- Tabuada do Número 4 ---
4 x 0 = 0
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

exercício 9 - conversor de moedas

Crie um programa que leia quanto dinheiro uma pessoa tem na carteira e mostre quantos dólares ela pode comprar.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Quanto dinheiro você tem na carteira? R$");
float dinheiro = teclado.nextFloat();

float dolaresHoje = (float) 5.81;
float dolares = dinheiro / dolaresHoje;

System.out.printf("Considerando o dólar em R$%.2f, Você consegue comprar $%.2f com o dinheiro que tem na carteira", dolaresHoje, dolares);
```

saída

Quanto dinheiro você tem na carteira? R\$150

Considerando o dólar em R\$5,81, Você consegue comprar \$25,82 com o dinheiro que tem na carteira

exercício 10 - pintando parede

Faça um programa que leia a largura e a altura de uma parede em metros, calcule a sua área e a quantidade de tinta necessária para pintá-la, sabendo que cada litro de tinta, pinta uma área de 2 metros quadrados.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite a largura da parede: ");
float largura = teclado.nextFloat();
System.out.print("Digite a altura da parede: ");
float altura = teclado.nextFloat();

float area = largura * altura;
float tinta = area / 2;

System.out.printf("A área da parede é %.2f metros quadrados", area);
System.out.printf("\nConsiderando que 1 litro de tinta de para pintar 2 metros quadrados, a quantidade de tinta necessária é %.2f litros", tinta);
```

saída

Digite a largura da parede: 5,2

Digite a altura da parede: 3

A área da parede é 15,60 metros quadrados

Considerando que 1 litro de tinta de para pintar 2 metros quadrados, será necessário 7,8

exercício 11 - calculando descontos

Faça um algoritmo que leia o preço de um produto e mostre seu novo preço, com 5% de desconto.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Qual o preço do produto? R$");
float preco = teclado.nextFloat();

float desconto = preco * (float) 0.95;

System.out.printf("O preço do produto com 5% de desconto é R$%.2f",desconto);
```

saída

```
Qual o preço do produto? R$45
O preço do produto com 5% de desconto é R$42,75
```

exercício 12 - reajuste salarial

Faça um algoritmo que leia o salário de um funcionário e mostre seu novo salário, com 15% de aumento.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Salário do funcionário: R$");
float salario = teclado.nextFloat();

float aumento = salario * (float) 1.15;

System.out.printf("Salário do funcionário com 15% de aumento: R$%.2f", aumento);
```

saída

Salário do funcionário: R\$1510,20

Salário do funcionário com 15% de aumento: R\$1736,73

exercício 13 - conversor de temperaturas

Escreva um programa que converta uma temperatura digitada em °C e converta em °F.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Temperatura em °C: ");
float tempC = teclado.nextFloat();

float tempF = (float) ((tempC * 1.8) + 32);

System.out.printf("Temperatura em °F: %.2f", tempF);
```

saída

Temperatura em °C: 20

Temperatura em °F: 68,00

exercício 14 - aluguel de carros

Escreva um programa que pergunte a quantidade de km percorridos por um carro alugado e a quantidade de dias pelos quais ele foi alugado. Calcule o preço a pagar, sabendo que o carro custa R\$60 por dia e R\$0,15 por km rodado.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("km percorridos: ");
float km = teclado.nextFloat();
System.out.print("Quantidade de dias: ");
int dias = teclado.nextInt();

float precokm = (float) (km * 0.15);
float precoDias = (float) (dias * 60);
float preco = precokm + precoDias;
```

```
System.out.printf("Você irá pagar R$%.2f pelos km rodados e R$%.2f pela quantidade de di  
  
saída  
km percorridos: 124  
Quantidade de dias: 3  
Você irá pagar R$18,60 pelos km rodados e R$180,00 pela quantidade de dias, dando um tot
```

▼ Exercícios Classes

exercício 15 - quebrando um número

Crie um programa que leia um número Real qualquer pelo teclado e mostre na tela a sua porção inteira.

```
comando  
import java.util.Scanner;  
  
Scanner teclado = new Scanner(System.in);  
System.out.print("Digite um número real: ");  
float num = teclado.nextFloat();  
  
int parteInt = (int) Math.floor(num);  
  
System.out.printf("Porção inteira desse número real: %d", parteInt);
```

```
saída  
Digite um número real: 145,142  
Porção inteira desse número real: 145
```

exercício 16 - catetos e hipotenusa

Faça um programa que leia o comprimento do cateto oposto e do cateto adjacente de um triângulo retângulo, calcule e mostre o comprimento da hipotenusa.

```
comando  
import java.util.Scanner;  
  
Scanner teclado = new Scanner(System.in);  
System.out.println("Vamos descobrir o comprimento da hipotenusa de um triângulo retângul
```

```
System.out.println("Iremos utilizar o teorema de pitágoras. ");
System.out.print("Digite o comprimento do cateto oposto: ");
float catOposto = teclado.nextFloat();
System.out.print("Digite o comprimento do cateto adjacente: ");
float catAdjacente = teclado.nextFloat();

float hipotenus = (float) (Math.sqrt(Math.pow(catOposto,2) + Math.pow(catAdjacente,2)));

System.out.printf("Comprimento da hipotenus: %.2f", hipotenus);
```

saída

Vamos descobrir o comprimento da hipotenus de um triângulo retângulo.

Iremos utilizar o teorema de pitágoras.

Digite o comprimento do cateto oposto: 3

Digite o comprimento do cateto adjacente: 4

Comprimento da hipotenus: 5,00

exercício 17 - seno, cosseno e tangente

Faça um programa que leia um ângulo qualquer e mostre na tela o valor do seno, cosseno e tangente desse ângulo.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Ângulo: ");
int angulo = teclado.nextInt();

float radianos = (float) Math.toRadians(angulo);
float seno = (float) Math.sin(radianos);
float cosseno = (float) Math.cos(radianos);
float tangente = (float) Math.tan(radianos);

System.out.printf("Valor do Seno: %.4f\n",seno);
System.out.printf("Valor do Cosseno: %.4f\n",cosseno);
System.out.printf("Valor do Tangente: %.4f\n",tangente);
```

saída

Ângulo: 45

Valor do Seno: 0,7071

Valor do Cosseno: 0,7071
Valor do Tangente: 1,0000

exercício 18 - sorteando um item na lista

Um professor quer sortear um dos seus quatro alunos para apagar o quadro. Faça um programa que ajude ele, lendo o nome deles e escrevendo o nome do escolhido.

comando

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

Scanner teclado = new Scanner(System.in);
Random random = new Random();
System.out.print("Aluno 1: ");
String aluno1 = teclado.nextLine();
System.out.print("Aluno 2: ");
String aluno2 = teclado.nextLine();
System.out.print("Aluno 3: ");
String aluno3 = teclado.nextLine();
System.out.print("Aluno 4: ");
String aluno4 = teclado.nextLine();

String[] alunos = {aluno1, aluno2, aluno3, aluno4};
int sorteio = random.nextInt(4);

System.out.println("Aluno Sorteado: " + alunos[sorteio]);
```

saída

```
Aluno 1: Wictor
Aluno 2: Francinaldo
Aluno 3: Matheus
Aluno 4: Gabriel
Aluno Sorteado: Matheus
```

exercício 19 - analisador de textos

Crie um programa que leia o nome completo de uma pessoa e mostre:
o nome com todas as letras maiúsculas.
o nome com todas as letras minúsculas.
quantas letras ao todo(sem considerar espaços).

quantas letras tem o primeiro nome.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o seu nome completo: ");
String nome = teclado.nextLine();

String maiusculo = nome.toUpperCase();
String minusculo = nome.toLowerCase();
int letras = nome.replace(" ", "").length();
int letraPrimeiro = nome.split(" ")[0].length();

System.out.println("Nome Maiúsculo: " + maiusculo);
System.out.println("Nome Minúsculo: " + minusculo);
System.out.println("Quantidade de Caracteres: " + letras);
System.out.println("Quantidade de Caracteres Primeiro Nome: " + letraPrimeiro);
```

saída

```
Digite o seu nome completo: Wictor Hugo Benevenutti
Nome Maiúsculo: WICTOR HUGO BENEVENUTTI
Nome Minúsculo: wictor hugo benevenutti
Quantidade de Caracteres: 21
Quantidade de Caracteres Primeiro Nome: 6
```

exercício 20 - separando dígitos de um número

Faça um programa que leia um número de 0 a 9999 e mostre na tela cada um dos dígitos separados.

Ex: Digite um número: 1834

unidade: 4

dezena: 3

centena: 8

milhar: 1

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número inteiro: ");
int num = teclado.nextInt();
```

```
int unidade = num % 10;
int dezena = (num / 10) % 10;
int centena = (num / 100) % 10;
int milhar = (num / 1000) % 10;

System.out.println("Unidade: " + unidade);
System.out.println("Dezena: " + dezena);
System.out.println("Centena: " + centena);
System.out.println("Milhar: " + milhar);
```

saída

```
Digite um número inteiro: 1435
Unidade: 5
Dezena: 3
Centena: 4
Milhar: 1
```

exercício 21 - verificando as primeiras letras de um texto

Crie um programa que leia o nome de uma cidade e diga se ela começa ou não com o nome "Santo".

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o nome da cidade: ");
String cidade = teclado.nextLine();

Boolean comparacao = cidade.toLowerCase().startsWith("santo");

System.out.println("Começa com Santo? " + comparacao);
```

saída

```
Digite o nome da cidade: Santo Antônio da Pratina
Começa com Santo? true
```

exercício 22 - procurando uma string dentro de outra

Crie um programa que leia o nome de uma pessoa e diga se ela tem "Silva" no nome.

```
comando
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o seu nome: ");
String nome = teclado.nextLine();

boolean comparacao = nome.contains("Silva");

System.out.println("Tem Silve no Nome? " + comparacao);
```

saída
Digite o seu nome: Wictor Hugo Benevenutti
Tem Silve no Nome? false

exercício 23 - primeira e última ocorrência de uma string

Faça um programa que leia uma frase pelo teclado e mostre:
quantas vezes aparece a letra "a".
em que posição ela aparece a primeira vez.
em que posição ela aparece a última vez.

```
comando
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Frase: ");
String frase = teclado.nextLine().toLowerCase();

int qtdeA = frase.length() - frase.replace("a", "").length();
int priA = frase.toLowerCase().indexOf('a') + 1;
int ultA = frase.toLowerCase().lastIndexOf('a') + 1;

System.out.println("Quantidade de A na frase: " + qtdeA);
System.out.println("Posição do primeiro A: " + priA);
System.out.println("Posição do último A: " + ultA);
```

saída
Frase: Faça um programa que leia uma frase pelo teclado e mostre:
Quantidade de A na frase: 8

Posição do primeiro A: 2

Posição do último A: 46

exercício 24 - primeiro e último nome de uma pessoa

Fala um programa que leia o nome completo de uma pessoa, mostrando em seguida o primeiro e o último nome separadamente.

Ex: Ana Maria de Souza

Primeiro = Ana

Último = Souza

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o seu nome completo: ");
String nome = teclado.nextLine();

String primeiro = nome.split(" ")[0];
String ultimo = nome.split(" ")[nome.split(" ").length - 1];

System.out.println("Primeiro = " + primeiro);
System.out.println("Último = " + ultimo);
```

saída

Digite o seu nome completo: Wictor Hugo Benevenutti

Primeiro = Wictor

Último = Benevenutti

▼ Exercícios Condições if e else

exercício 25 - jogo da adivinhação

Escreva um programa que faça o computador "pensar" em um número inteiro entre 0 e 5 e peça para o usuário tentar descobrir qual foi o número escolhido pelo computador. O programa deverá escrever na tela se o usuário venceu ou perdeu.

comando

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

```
Scanner teclado = new Scanner(System.in);
Random random = new Random();
System.out.print("Tente adivinhar o número que eu pensei de 0 a 5: ");
int num = teclado.nextInt();

int sorteado = random.nextInt(6);

System.out.println("O número sorteado foi " + sorteado);
if (num == sorteado) {
    System.out.println("Você Venceu! :)");
} else {
    System.out.println("Você Perdeu! :(");
}
```

saída

```
Tente adivinhar o número que eu pensei de 0 a 5: 4
O número sorteado foi 0
Você Perdeu! :(
```

exercício 26 - radar eletrônico

Escreva um programa que leia a velocidade de um carro. Se ele ultrapassar 80km/h, mostre uma mensagem dizendo que ele foi multado.

A multa vai custar R\$7,00 por cada km acima do limite.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Velocidade do carro: ");
float velocidade = teclado.nextFloat();

if (velocidade < 80) {
    System.out.println("Dentro do limite de velocidade!");
} else {
    float multa = (velocidade - 80) * 7;
    System.out.println("Você ultrapassou o limite de velocidade!");
    System.out.println("Multa de R$" + multa);
}
```

saída

```
Velocidade do carro: 90
```

Você ultrapassou o limite de velocidade!
Multa de R\$70.0

exercício 27 - par ou ímpar?

Crie um programa que leia um número inteiro e mostre na tela se ele é PAR ou IMPAR.

comando

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

```
Scanner teclado = new Scanner(System.in);
System.out.print("Digite um número inteiro: ");
int num = teclado.nextInt();

if ((num % 2) == 0) {
    System.out.println("É um número PAR!");
} else {
    System.out.println("É um número IMPAR!");
}
```

saída

```
Digite um número inteiro: 9
É um número IMPAR!
```

exercício 28 - custo da viagem

Desenvolva um programa que pergunte a distância de uma viagem em km. Calcule o preço da passagem, cobrando 0,50 reais por km para viagens de até 200km e 0,45 reais para viagens mais longas.

comando

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

```
Scanner teclado = new Scanner(System.in);
System.out.print("Distância da Viagem: ");
float distancia = teclado.nextFloat();

if (distancia <= 200) {
    System.out.println("Preço da passagem: R$" + distancia * 0.50);
} else {
    System.out.println("Preço da passagem: R$" + distancia * 0.45);
}
```

saída
Distância da Viagem: 180
Preço da passagem: R\$90.0

exercício 29 - ano bissexto

Faça um programa que leia um ano qualquer e mostre se ele é BISSEXTO.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite um Ano: ");
int ano = teclado.nextInt();
if (((ano % 4) == 0) && ((ano % 100) != 0)) || ((ano % 400) == 0)) {
    System.out.println("O Ano " + ano + " é Bissesto!");
} else {
    System.out.println("O Ano " + ano + " não é Bissesto!");
}
```

saída

```
Digite um Ano: 2001
O Ano 2001 não é Bissesto!
```

exercício 30 - maior ou menor valores

Faça um programa que leia três números e mostre qual é o maior e qual é o menor.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.println("Digite 3 Números.");
System.out.print("Número 1: ");
int num1 = teclado.nextInt();
System.out.print("Número 2: ");
int num2 = teclado.nextInt();
System.out.print("Número 3: ");
int num3 = teclado.nextInt();

int menor = 0; int maior = 0;

if ((num1 <= num2) && (num1 <= num3)) {
```

```

menor = num1;
} else if ((num2 <= num1) && (num2 <= num3)) {
    menor = num2;
} else {
    menor = num3;
}
if ((num1 >= num2) && (num1 >= num3)) {
    maior = num1;
} else if ((num2 >= num1) && (num2 >= num3)) {
    maior = num2;
} else {
    maior = num3;
}

System.out.println("Menor Número: " + menor);
System.out.println("Maior Número: " + maior);

```

saída

```

Digite 3 Números.
Número 1: 4
Número 2: 15
Número 3: 2
Menor Número: 2
Maior Número: 15

```

exercício 31 - aumentos múltiplos

Escreva um programa que pergunte o salário de um funcionário e calcule o valor do seu aumento. Para salários superiores a R\$1250,00, calcule um aumento de 10%. Para os inferiores ou iguais, o aumento é de 15%.

comando

```

import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.print("Digite o seu salário: R$");
double salario = teclado.nextFloat();

if (salario > 1250) {
    System.out.println("Com 10% de aumento o seu novo salário será R$" + Math.round(salario * 1.1));
} else {
    System.out.println("Com 15% de aumento o seu novo salário será R$" + Math.round(salario * 1.15));
}

```

saída

Digite o seu salário: R\$1400

Com 10% de aumento o seu novo salário será R\$1540

exercício 32 - analisando triângulo

Desenvolva um programa que leia o comprimento de três retas e diga ao usuário se elas podem ou não formar um triângulo.

comando

```
import java.util.Scanner;

Scanner teclado = new Scanner(System.in);
System.out.println("Digite o comprimento de 3 retas.");
System.out.print("Reta 1: ");
float r1 = teclado.nextFloat();
System.out.print("Reta 2: ");
float r2 = teclado.nextFloat();
System.out.print("Reta 3: ");
float r3 = teclado.nextFloat();

if (((r1 < (r2 + r3)) && (r2 < (r1 + r3))) && (r3 < (r1 + r2))) {
    System.out.println("As retas podem formar um triângulo!");
} else {
    System.out.println("As retas não podem formar um triângulo!");
}
```