Módulo 0: Revisão de Java - Mergulhando nos Fundamentos

💡 **EEEP Amélia Figueiredo de Lavor**

Curso: Técnico em Informática - Turma 2º Ano - 2025.1

Professor: @jhordanleandro

💡 **Objetivo:** Antes de embarcarmos na aventura da Programação Orientada a Objetos, vamos fortalecer nossos conhecimentos sobre os pilares da linguagem Java. Este módulo é como nossa preparação para uma maratona: revisaremos a sintaxe básica, os tipos de dados, operadores, estruturas de controle e métodos - tudo essencial para construirmos programas robustos e eficientes.

▼ 1. Sintaxe Básica: A Gramática da Linguagem Java

Assim como em qualquer idioma, a linguagem Java possui suas próprias regras gramaticais, que chamamos de sintaxe. Dominar a sintaxe é essencial para escrevermos código que o computador possa entender e executar.

Estrutura de um Programa Java: Imagine um programa Java como uma receita de bolo. A estrutura básica inclui:

- **Declaração de Classes:** A receita em si, com o nome do bolo (classe) e os ingredientes (atributos).
- **Método Principal (`main`):** O passo a passo da receita, onde a mágica acontece.
- **Importação de Bibliotecas:** Semelhante a usar ingredientes prontos, como fermento ou cobertura, para facilitar o processo.
- **Uso de Ponto e Vírgula (`;`):** Como a pontuação em uma frase, indica o fim de uma instrução.
- **Blocos de Código (`{}`):** As seções da receita, que agrupam instruções relacionadas.

```
import java.util.Scanner; // Importa a biblioteca Scanner

public class OlaMundo { // Declara a classe OlaMundo

    public static void main(String args) { // Método principal
        Scanner entrada = new Scanner(System.in); // Cria um objeto Scanner
        System.out.print("Digite seu nome: "); // Imprime na tela
        String nome = entrada.nextLine(); // Lê a entrada do usuário
        System.out.println("Olá, " + nome + "!"); // Imprime a saudação
    }
}
```

Comentários: São como anotações na receita, úteis para lembrarmos de detalhes ou explicarmos o código para outras pessoas.

- **Comentários de Linha (`//`)**: Explicam algo em uma única linha.
- **Comentários de Bloco (`/* */`)**: Úteis para comentários mais longos.

```
// Este é um comentário de linha

/*
Este é um comentário
de bloco
*/
```

Identificadores: Os nomes que damos para as classes, métodos e variáveis.

- **Regras:** Devem começar com letra, `_` ou `$`, e podem conter letras, números e símbolos.
- **Convenções:** Use nomes descritivos e siga o padrão CamelCase (ex: `meuNome`, `calcularSalario`).

Palavras-chave: Palavras reservadas da linguagem, que não podemos usar como identificadores (ex: `public`, `class`, `static`, `void`, `int`, `if`, `else`, `for`, `while`).

▼ 2. Tipos de Dados Primitivos: Os Ingredientes do Bolo

Assim como uma receita precisa de diferentes tipos de ingredientes, a linguagem Java oferece diversos tipos de dados para armazenarmos informações.

Tipos Numéricos:

- **Inteiros:** `byte` (1 byte), `short` (2 bytes), `int` (4 bytes), `long` (8 bytes).
- **Ponto Flutuante:** `float` (4 bytes, precisão simples) e `double` (8 bytes, precisão dupla).

```
int idade = 25;
double preco = 19.99;
```

Tipo Caractere (`char`): Armazena um único caractere Unicode (ex: 'A', '?', 'π').

```
char inicial = 'J';
```

Tipo Booleano (`boolean`): Representa valores lógicos `true` (verdadeiro) ou `false` (falso).

```
boolean ligado = true;
```

Variáveis: São como recipientes que armazenam os dados.

- **Declaração:** Tipo + nome (ex: `int quantidade;`).
- **Atribuição:** `quantidade = 10;`.
- **Constantes:** Variáveis imutáveis, declaradas com `final` (ex: `final double PI = 3.14159;`).

▼ 3. Operadores: As Ferramentas do Chef

Os operadores são como as ferramentas que usamos para manipular os ingredientes da receita.

Operadores Aritméticos:

- `+` (adição), `-` (subtração), `*` (multiplicação), `/` (divisão), `%` (módulo/resto da divisão).

```
int soma = 10 + 5; // 15
int resto = 10 % 3; // 1
```

Operadores Relacionais: Comparam valores e retornam um resultado booleano.

- `==` (igual a), `!=` (diferente de), `>` (maior que), `<` (menor que), `>=` (maior ou igual a), `<=` (menor ou igual a).

```
boolean igual = (5 == 5); // true
boolean maior = (10 > 5); // true
```

Operadores Lógicos: Combinam valores booleanos.

- `&&` (AND lógico - "e"), `||` (OR lógico - "ou"), `!` (NOT lógico - "não").

```
boolean resultado = (true && false); // false
boolean resultado2 = (true || false); // true
```

Operadores de Atribuição: Atribuem valores a variáveis.

- `=` (atribuição simples), `+=` (adição e atribuição), `-=` (subtração e atribuição), etc.

```
int x = 10;
x += 5; // x agora vale 15
```

Operadores de Incremento e Decremento:

- `++` (incrementa em 1), `--` (decrementa em 1).

```
int i = 5;
i++; // i agora vale 6
```

▼ 4. Estruturas de Controle de Fluxo: Controlando o Fluxo da Receita

As estruturas de controle de fluxo permitem que o programa tome decisões e execute blocos de código específicos, como se estivéssemos controlando o fluxo da receita.

Estrutura Condicional `if-else`**:** Permite executar um bloco de código se uma condição for verdadeira, e outro bloco caso contrário.

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

Estrutura Condicional `switch`**:** Útil para escolher entre múltiplas opções, com base no valor de uma variável.

```
switch (diaSemana) {
case 1:
System.out.println("Domingo");
break;
case 2:
System.out.println("Segunda");
break;
//... outros casos
default:
System.out.println("Dia inválido");
}
```

▼ 5. Estruturas de Repetição: Repetindo Etapas da Receita

As estruturas de repetição permitem que o programa execute um bloco de código várias vezes, como se estivéssemos repetindo etapas da receita.

Laço `for`**:** Ideal para quando sabemos quantas vezes queremos repetir o código.

```
for (int i = 0; i < 10; i++) {
System.out.println(i); // Imprime os números de 0 a 9
}
```

Laço `while`**:** Repete o código enquanto uma condição for verdadeira.

```
int contador = 0;
while (contador < 5) {
System.out.println("Contador: " + contador);
contador++;
}
```

Laço `do-while`**:** Semelhante ao `while`, mas garante que o código seja executado pelo menos uma vez.

```
int numero;
do {
System.out.print("Digite um número positivo: ");
numero = entrada.nextInt();
} while (numero <= 0);
```

▼ 6. Arrays: Organizando os Ingredientes em uma Lista

Os arrays são como listas que nos permitem armazenar vários valores do mesmo tipo, como se estivéssemos organizando os ingredientes da receita em uma lista.

- **Declaração:** `tipo nomeArray;` (ex: `int numeros;`).
- **Inicialização:** `numeros = new int;` (cria um array de 5 inteiros).

- **Acesso:** `numeros = 10;` (atribui o valor 10 ao primeiro elemento do array).
- **Iteração:** Use laços `for` para percorrer os elementos do array.

```
int idades = {25, 30, 18, 42};
for (int i = 0; i < idades.length; i++) {
System.out.println("Idade " + (i + 1) + ": " + idades[i]);
}
```

▼ 7. Métodos: Dividindo a Receita em Tarefas

Os métodos são como sub-receitas dentro da receita principal, que nos permitem dividir o código em tarefas menores e mais organizadas.

- **Declaração:** `tipoRetorno nomeMetodo(parametros) { corpo }` (ex: `int somar(int a, int b) { return a + b; }`).
- **Chamada:** `int resultado = somar(5, 3);`.
- **Escopo:** Variáveis declaradas dentro de um método só existem dentro dele.
- **Sobrecarga:** Podemos ter vários métodos com o mesmo nome, mas com parâmetros diferentes.

```
public class Calculadora {
    public int somar(int a, int b) {
        return a + b;
    }

    public double somar(double a, double b) {
        return a + b;
    }
}
```

Exercícios:

1. Crie um programa que leia dois números do usuário e imprima a soma, subtração, multiplicação e divisão deles.
2. Escreva um programa que leia um número inteiro e verifique se ele é par ou ímpar.
3. Crie um programa que leia 10 números do usuário e armazene-os em um array. Em seguida, imprima o maior e o menor número do array.
4. Escreva um método que receba um array de inteiros como parâmetro e retorne a soma de todos os elementos do array.

> **OBS.: ENVIAR O ARQUIVO COM AS RESPOSTAS DOS EXERCÍCIOS!**