

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL

AULA 03

LINGUAGEM DE PROGRAMAÇÃO 2
JAVA



PROF. JANIHERYSON FELIPE

CONTEÚDO DESSA AULA

- INTRODUZIR A ORIENTAÇÃO A OBJETOS EM JAVA**
- CRIAR OBJETOS E CLASSES EM JAVA E CONSTRUTORES;**
- METODOS GET/SET**
- DISCUSSÕES E DÚVIDAS GERAIS.**

PARADIGMAS

ESTRUTURADO

VS

ORIENTAÇÃO A OBJETOS

PARADIGMAS DE PROGRAMAÇÃO

Programação estruturada (procedural):

- Descreve a computação como ações ou comandos que mudam o estado (variáveis) de um programa fonte.
- IF, FOR, WHILE, etc...
- Baseado no conceito de chamadas a procedimentos

PARADIGMAS DE PROGRAMAÇÃO

Programação orientada a objetos:

- A orientação a objetos é um paradigma de análise, projeto e programação de sistemas de software.
- Baseado na composição e interação entre diversas unidades de software chamadas de objetos

ORIENTAÇÃO A OBJETOS

- Surge nos anos 60~70 com a linguagem Smalltalk.
- Se espalhou para diversas linguagens: C++, C#, Java, Python, Ruby, Kotlin, etc.
- “Uma nova maneira de pensar os problemas utilizando conceitos do Mundo Real. O componente fundamental é o objeto que combina estrutura e comportamento em uma única entidade”.

ORIENTAÇÃO A OBJETOS

- Reuso do código
- Reflete o mundo real
- Facilita a manutenção no código



ORIENTAÇÃO A OBJETOS

Observações:

- Muitas **instâncias** podem ser criadas a partir de uma **única classe**.
- Um objeto tem atributos: **valores armazenados** em **campos**.
- A classe define quais campos um objeto tem, mas cada objeto armazena seu **próprio conjunto de valores** (o estado do objeto).

ORIENTAÇÃO A OBJETOS

Métodos são parecidos com funções:

- Recebem parâmetros;
- Possuem um valor de retorno.



▼ Atributos

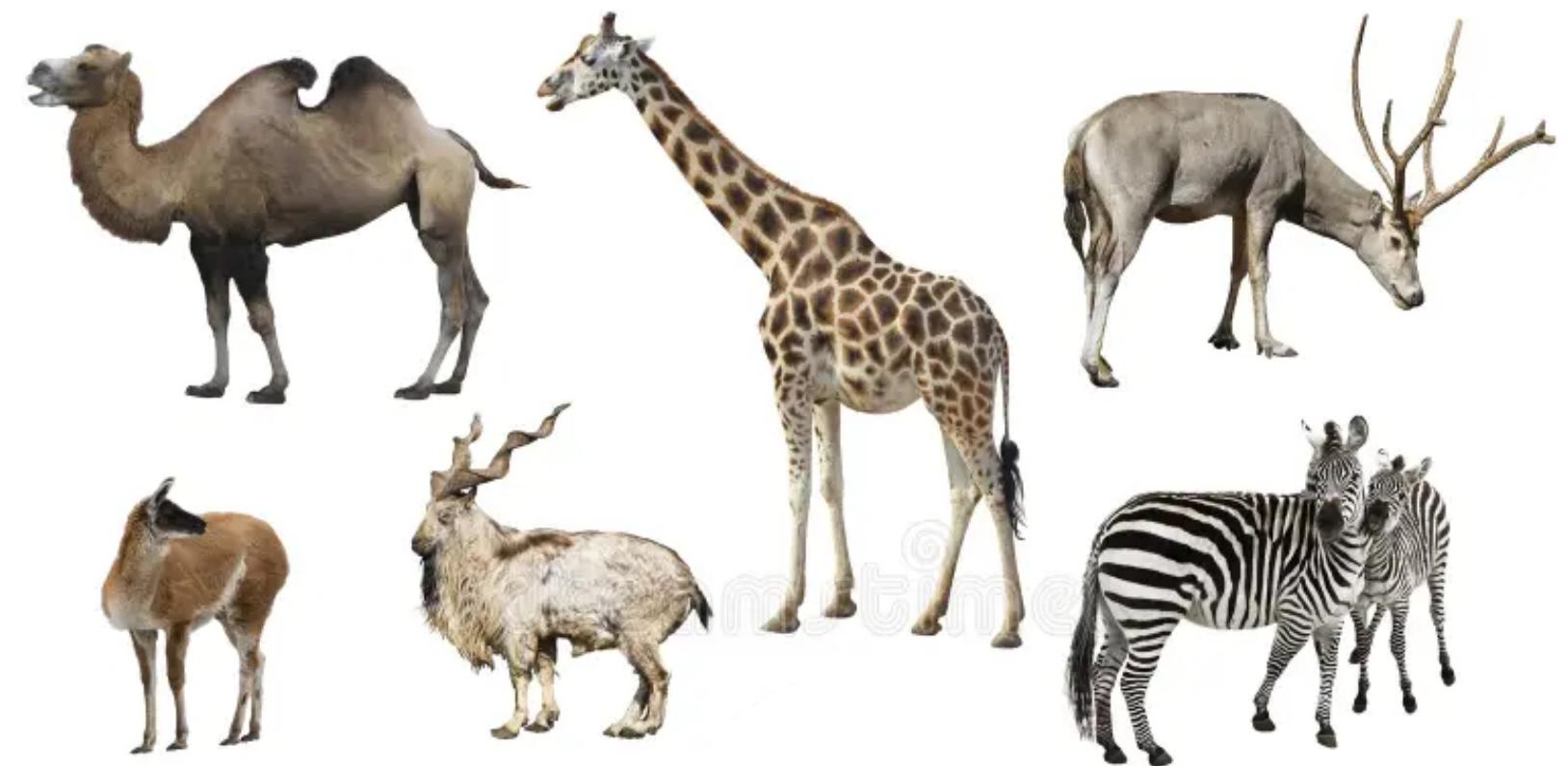
- ▼ Potência: 500cc
- ▼ Marca: Honda
- ▼ Ano: 1998
- ▼ Velocidade: 100km/h

▼ Métodos

- ▼ Acelerar
- ▼ Frear
- ▼ Abastecer
- ▼ Ver velocidade

CLASSES

- Descrição de um conjunto de objetos ou elementos semelhantes.
- Ideia de agupamento de objetos com características em comum.



O QUE É UMA CLASSE...

- Nome da classe
- Conjunto de atributos (descrição)
- Conjunto de métodos (comportamentos)



OBJETO: ROTTWEILER



Nome da classe: Cachorro

Conjunto de atributos:

- Tamanho max: 68 cm;
- Peso: 50 kg;
- Espectativa de vida: 10 anos;
- Energia: média;
- Adestramento: alto;
- Inteligencia: alta;
- Agressividade: alta;

OBJETO: PINSCHER

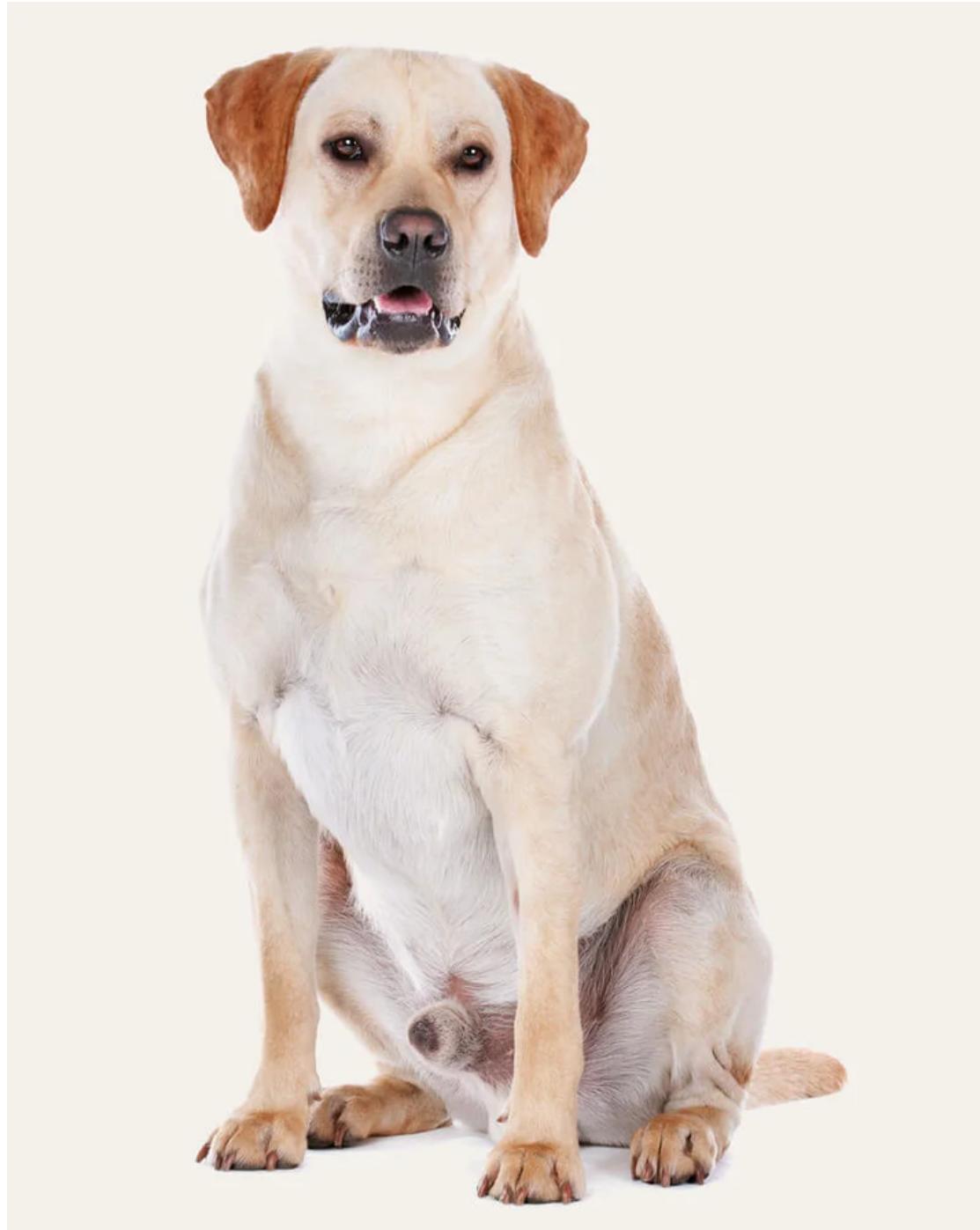


Nome da classe: Cachorro

Conjunto de atributos:

- Tamanho max: 30 cm;
- Peso: 4 kg;
- Espectativa de vida: 15 anos;
- Energia: alta;
- Adestramento: baixo;
- Inteligencia: media-alta;
- Agressividade: alta;

OBJETO: LABRADOR



Nome da classe: Cachorro

Conjunto de atributos:

- Tamanho max: 57 cm;
- Peso: 30 kg;
- Espectativa de vida: 12 anos;
- Energia: média;
- Adestramento: alto;
- Inteligencia: alta;
- Agressividade: baixa;

CONJUNTO DE MÉTODOS

- latir
- correr
- proteger
- comer
- passear
- destruirSandalias
- fazerBagunça



ABSTRAÇÃO E MODULARIZAÇÃO

Abstração:

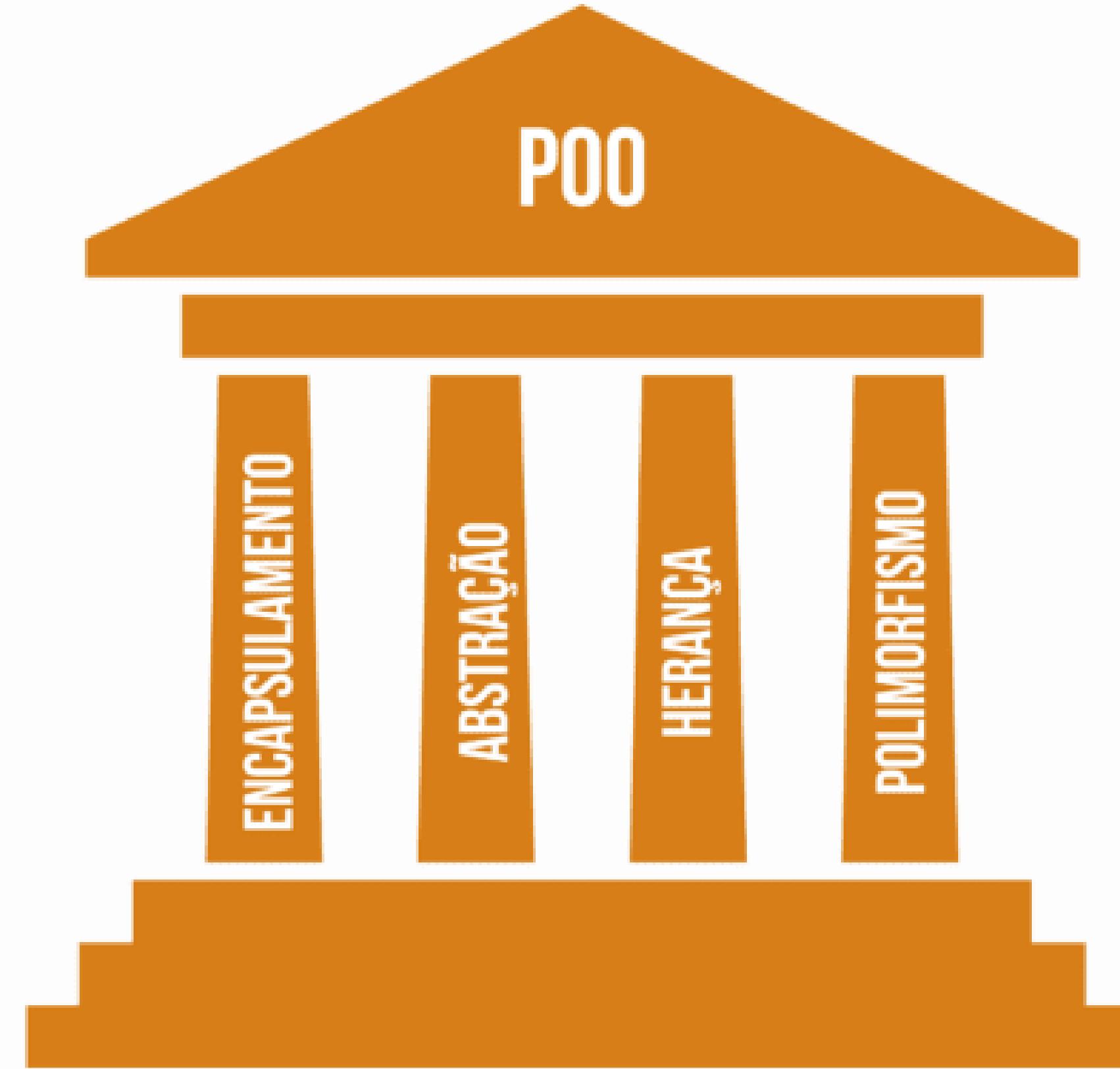
- Capacidade de ignorar detalhes de partes para focalizar a atenção em um nível mais elevado de um problema.

Modularização:

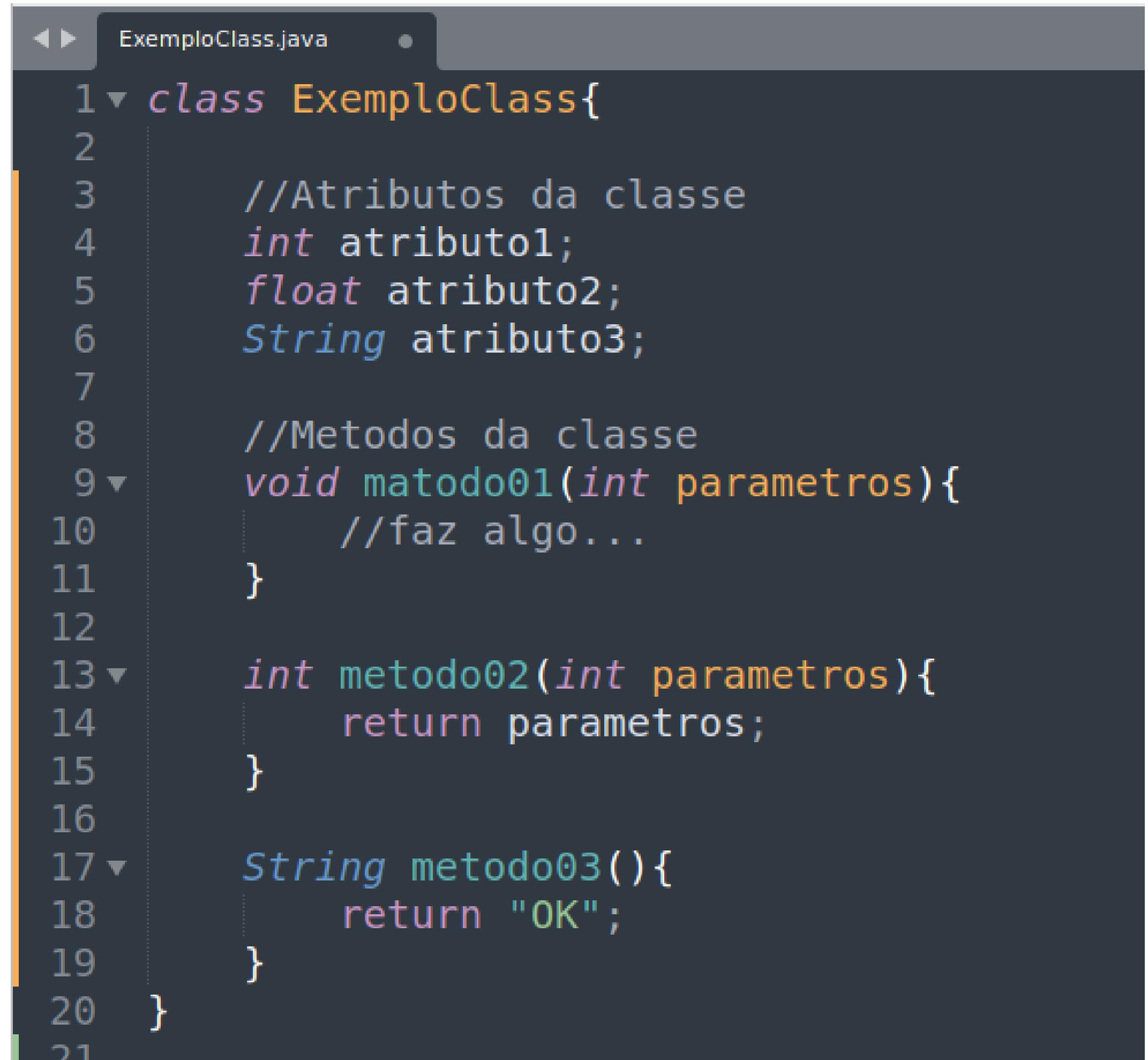
- Processo de dividir um todo em partes bem definidas, que podem ser construídas e examinadas separadamente, e que interagem.
- Exemplo: carro.
 - Motor, assentos, volante, freios, pneus.
 - Motor: cilindros, velas, eixos, injeção, eletrônica.

PILARES DA ORIENTAÇÃO A OBJETOS

- Encapsulamento
- Abstração
- Herança
- Polimorfismos



FORMA GERAL DE UMA CLASSE



```
ExemploClass.java

1 ▼ class ExemploClass{
2
3     //Atributos da classe
4     int atributo1;
5     float atributo2;
6     String atributo3;
7
8     //Metodos da classe
9 ▼     void metodo01(int parametros){
10         //faz algo...
11     }
12
13 ▼     int metodo02(int parametros){
14         return parametros;
15     }
16
17 ▼     String metodo03(){
18         return "OK";
19     }
20 }
```

**CRIE A CLASSE
CARRO COM
OS ATRIBUTOS**



- cor
- marca
- modelo
- ano de fabricação
- numero de portas
- consumo de combustivel

**ADICIONE À
CLASSE CARRO
OS MÉTODOS**



- ligar
- desligar
- acelerar
- frear
- parar

CRIAÇÃO DE OBJETOS DE UMA CLASSE



CRIAÇÃO DE OBJETOS DE UMA CLASSE



- cor: prata
- marca: Lamborghini
- modelo: Aventador
- ano de fabricação: 2011
- numero de portas: 2
- consumo: 3 km/l
- preço: 8.000.000,00 (R\$)

CRIAÇÃO DE OBJETOS DE UMA CLASSE



- cor: prata
- marca: Fiat
- modelo: Uno
- ano de fabricação: 2009
- numero de portas: 4
- consumo: 13 km/l
- preço: 17.930 R\$

CRIAÇÃO DE OBJETOS DE UMA CLASSE



- cor: Azul
- marca: Volkswagen
- modelo: Brasilia
- ano de fabricação: 1978
- numero de portas: 2
- consumo: 11 km/l
- preço: 32.930 R\$

CONSTRUTOR DE UMA CLASSE

Em poo, um construtor é um método especial dentro de uma classe que é **chamado automaticamente** quando um objeto da classe é **criado**. Sua principal responsabilidade é **inicializar** os atributos do objeto e realizar outras tarefas de inicialização necessárias.



CONSTRUTOR DE UMA CLASSE

```
class Carro{  
    String modelo;  
    String marca;  
    double capacidadeTanque;  
  
    public Carro(String modelo, String marca){  
        this.marca = marca;  
        this.modelo = modelo;  
        this.capacidadeTanque = 0;  
    }  
}
```

CONSTRUTOR DE UMA CLASSE

```
public class Main {  
    public static void main(String[] args) {  
        Carro uno = new Carro("Fiat", "uno");  
        System.out.println(uno.modelo);  
        System.out.println(uno.marca);  
        System.out.println(uno.capacidadeTanque);  
    }  
}
```

SOBRECARGA DE CONSTRUTOR

```
public Carro(String modelo, String marca){  
    this.marca = marca;  
    this.modelo = modelo;  
    this.capacidadeTanque = 0;  
}  
  
public Carro(){  
    System.out.println("Objeto vazio criado!");  
}
```

METODOS GET/SET

Os métodos getter e setter são comumente usados para **acessar e modificar** os valores dos atributos de uma classe, respectivamente. Esses métodos são uma parte fundamental do **encapsulamento**, um conceito-chave na programação orientada a objetos que visa proteger os detalhes internos de uma classe e fornecer um controle mais preciso sobre como os atributos são acessados e modificados.

METODOS GET/SET

```
public String getMarca() {  
    return marca;  
}  
  
public void setMarca(String marca) {  
    this.marca = marca;  
}  
  
public double getCapacidadeTanque() {  
    return capacidadeTanque;  
}  
  
public void setCapacidadeTanque(double capacidadeTanque) {  
    this.capacidadeTanque = capacidadeTanque;  
}
```

