**Classes e Objetos em Java**

**1. Introdução**

* **Classe**: Uma **classe** é um conceito central na programação orientada a objetos (POO). Ela pode ser vista como um molde ou um modelo que define as propriedades e comportamentos que os objetos daquela classe terão. Em outras palavras, uma classe serve como uma estrutura que agrupa dados e métodos que operam sobre esses dados.
* **Objeto**: Um **objeto** é uma instância de uma classe. Quando você cria um objeto, está criando uma entidade específica que possui os atributos e métodos definidos na classe. Cada objeto pode ter valores diferentes para os atributos, mesmo que todos compartilhem a mesma estrutura da classe.

**2. Definindo uma Classe**

Uma classe é definida usando a palavra-chave class. A sintaxe básica é a seguinte:

public class NomeDaClasse {

// Atributos (propriedades)

tipo atributo1;

tipo atributo2;

// Construtor

public NomeDaClasse(tipo parametro1, tipo parametro2) {

this.atributo1 = parametro1;

this.atributo2 = parametro2;

}

// Métodos

public void metodoExemplo() {

// Lógica do método

}

}

**3. Criando um Objeto**

Para criar um objeto, você usa a palavra-chave “new” seguida do construtor da classe:

NomeDaClasse objeto = new NomeDaClasse(valor1, valor2);

**4. Exemplo Prático**

// Definição da classe

public class Carro {

// Atributos

private String modelo;

private int ano;

// Construtor

public Carro(String modelo, int ano) {

this.modelo = modelo;

this.ano = ano;

}

// Método para exibir informações

public void mostrarInfo() {

System.out.println("Modelo: " + modelo + ", Ano: " + ano);

}

}

// Uso da classe

public class Main {

public static void main(String[] args) {

Carro meuCarro = new Carro("Fusca azul", 1970);

meuCarro.mostrarInfo();

}

}

**5. Modificadores de Acesso**

* **public**: Acesso total.
* **private**: Acesso restrito à própria classe.
* **protected**: Acesso à classe e suas subclasses.

**6. Métodos Getters e Setters**

Para manipular atributos privados, utiliza-se getters e setters:

public String getModelo() {

return modelo;

}

public void setModelo(String modelo) {

this.modelo = modelo;

}

**7. Herança**

Java suporta herança, permitindo que uma classe herde atributos e métodos de outra:

public class Veiculo {

// Atributos e métodos comuns

}

public class Carro extends Veiculo {

// Atributos e métodos específicos

}

**8. Polimorfismo**

Permite que métodos em classes diferentes possam ser chamados através de um mesmo nome. É implementado por meio de sobrecarga (overloading) e sobrescrita (overriding).

**9. Encapsulamento**

É o princípio de esconder os detalhes internos de uma classe e expor apenas o que é necessário através de métodos públicos.

**Conclusão**

Classes e objetos são fundamentais na programação orientada a objetos em Java. Compreender esses conceitos permitirá que você desenvolva aplicações mais organizadas e eficientes.