# Herança em Java

A herança é um dos pilares da programação orientada a objetos. Em Java, é implementada usando a palavra-chave `extends`. Isso permite que uma classe (subclasse) herde atributos e métodos de outra classe (superclasse), promovendo a reutilização de código e estabelecendo relações entre classes.

## Principais Conceitos:

**1. Superclasse e Subclasse:**
- A superclasse é a classe da qual outra classe (subclasse) herda.
- A subclasse é a classe que herda da superclasse.

**2. Extensão de Funcionalidades:**
- A subclasse pode estender a funcionalidade da superclasse adicionando novos atributos e métodos ou sobrescrevendo os métodos existentes.
- Isso promove a reutilização de código e a organização hierárquica de classes.

**3. Modificadores de Acesso:**
- Os modificadores de acesso (`public`, `protected`, `private`) controlam a visibilidade dos membros da superclasse na subclasse.
- Uma subclasse pode acessar os membros protegidos e públicos da superclasse, mas não os membros privados.

**Exemplo de Código:**

```java
// Superclasse
public class Veiculo {
    protected String marca;
    protected String modelo;

    public void acelerar() {
        System.out.println("Veículo acelerando");
    }
}

// Subclasse
public class Carro extends Veiculo {
    private int ano;

    public void setAno(int ano) {
        this.ano = ano;
    }
}
```

# Polimorfismo em Java

O polimorfismo é outro conceito fundamental da programação orientada a objetos. Ele permite que um objeto possa ser tratado de várias formas, dependendo do contexto em que é usado.

**Tipos de Polimorfismo em Java:**

**1. Polimorfismo de Subtipo (Subtyping):**

- Permite que um objeto de uma subclasse seja tratado como um objeto de sua superclasse.
- Isso facilita o uso de uma interface mais genérica para interagir com diferentes tipos de objetos.

**2. Polimorfismo Paramétrico (Generics):**

- Permite que classes e métodos sejam parametrizados por tipos, permitindo maior flexibilidade e segurança de tipos.

**Benefícios do Polimorfismo:**

- **Flexibilidade**: Permite escrever código genérico que pode lidar com diferentes tipos de objetos.
- **Reutilização de Código**: Facilita a escrita de código mais limpo e conciso, evitando repetição.
- **Extensibilidade**: Permite adicionar novos comportamentos ou tipos de objetos sem modificar o código existente.

**Exemplo de Código:**

```
public class Programa {
    public static void main(String[] args) {
        Veiculo veiculo = new Carro(); // Polimorfismo de subtipo
        veiculo.acelerar(); // Chama o método da subclasse

        List<Animal> animais = new ArrayList<>(); // Polimorfismo paramétrico
        animais.add(new Cachorro());
        animais.add(new Gato());
        for (Animal animal : animais) {
            animal.emitirSom(); // Chama o método específico de cada subclasse
        }
    }
}
```

## Conclusão

A herança e o polimorfismo são conceitos poderosos que permitem criar hierarquias de classes flexíveis e escrever código mais reutilizável e extensível em Java. Ao entender e aplicar esses conceitos corretamente, os desenvolvedores podem criar sistemas mais robustos e fáceis de manter.