Introdução à Programação Orientada a Objetos em Java

1. Introdução à Programação Orientada a Objetos

Conceitos Básicos

- Classe: Modelo ou plano de um objeto que define um conjunto de atributos e métodos.
- Objeto: Instância de uma classe que possui estados e comportamentos definidos pela classe.
- Herança: Mecanismo onde uma classe pode herdar atributos e métodos de outra classe.
- Polimorfismo: Capacidade de um método ter diferentes comportamentos baseados no objeto que está chamando o método.
- Encapsulamento: Técnica de esconder os detalhes de implementação dos atributos e métodos de uma classe, expondo apenas o necessário.
- Abstração: Simplificação da complexidade através da definição de classes que representam conceitos genéricos.

Introdução à Programação Orientada a Objetos em Java

2. Classes e Objetos

Definindo uma Classe

```java
public class Carro {
    String marca;
    String modelo;
    int ano;

    public Carro(String marca, String modelo, int ano) {
        this.marca = marca;
        this.modelo = modelo;
        this.ano = ano;
    }

    public void exibirDetalhes() {
        System.out.println("Marca: " + marca);
        System.out.println("Modelo: " + modelo);
        System.out.println("Ano: " + ano);
    }
}
```

Criando Objetos

```java
public class Principal {
```

```
    public static void main(String[] args) {
        Carro carro1 = new Carro("Toyota", "Corolla", 2020);
        Carro carro2 = new Carro("Honda", "Civic", 2021);
        carro1.exibirDetalhes();
        carro2.exibirDetalhes();
    }
}
```

Introdução à Programação Orientada a Objetos em Java

3. Herança

Definindo Herança

```java
class Veiculo {
    String marca;
    String modelo;

    public Veiculo(String marca, String modelo) {
        this.marca = marca;
        this.modelo = modelo;
    }

    public void exibirDetalhes() {
        System.out.println("Marca: " + marca);
        System.out.println("Modelo: " + modelo);
    }
}

class Carro extends Veiculo {
    int ano;

    public Carro(String marca, String modelo, int ano) {
        super(marca, modelo);
        this.ano = ano;
    }
```

```
    @Override
    public void exibirDetalhes() {
        super.exibirDetalhes();
        System.out.println("Ano: " + ano);
    }
}

public class Principal {
    public static void main(String[] args) {
        Carro carro = new Carro("Toyota", "Corolla", 2020);
        carro.exibirDetalhes();
    }
}
```

Introdução à Programação Orientada a Objetos em Java

4. Polimorfismo

Polimorfismo em Ação

```java
class Animal {
    public void emitirSom() {
        System.out.println("O animal faz um som.");
    }
}

class Cao extends Animal {
    @Override
    public void emitirSom() {
        System.out.println("O cão late.");
    }
}

class Gato extends Animal {
    @Override
    public void emitirSom() {
        System.out.println("O gato mia.");
    }
}

public class Principal {
    public static void main(String[] args) {
```

```
        Animal meuCao = new Cao();
        Animal meuGato = new Gato();
        meuCao.emitirSom();
        meuGato.emitirSom();
    }
}
```

Introdução à Programação Orientada a Objetos em Java

5. Encapsulamento

Definindo Encapsulamento

```java
public class ContaBancaria {
    private double saldo;

    public ContaBancaria(double saldoInicial) {
        this.saldo = saldoInicial;
    }

    public double getSaldo() {
        return saldo;
    }

    public void depositar(double valor) {
        if (valor > 0) {
            saldo += valor;
        }
    }

    public void sacar(double valor) {
        if (valor > 0 && saldo >= valor) {
            saldo -= valor;
        }
    }
```

```
}


public class Principal {
    public static void main(String[] args) {
        ContaBancaria minhaConta = new ContaBancaria(1000.00);
        minhaConta.depositar(500.00);
        System.out.println("Saldo após depósito: " + minhaConta.getSaldo());
        minhaConta.sacar(200.00);
        System.out.println("Saldo após saque: " + minhaConta.getSaldo());
    }
}
```

6. Abstração

Definindo Abstração

```java
abstract class Forma {
    public abstract double calcularArea();
}

class Circulo extends Forma {
    private double raio;

    public Circulo(double raio) {
        this.raio = raio;
    }

    @Override
    public double calcularArea() {
        return Math.PI * raio * raio;
    }
}

class Retangulo extends Forma {
    private double largura;
    private double altura;

    public Retangulo(double largura, double altura) {
```

```
        this.largura = largura;
        this.altura = altura;
    }

    @Override
    public double calcularArea() {
        return largura * altura;
    }
}

public class Principal {
    public static void main(String[] args) {
        Forma circulo = new Circulo(5.0);
        Forma retangulo = new Retangulo(4.0, 6.0);
        System.out.println("Área do círculo: " + circulo.calcularArea());
        System.out.println("Área do retângulo: " + retangulo.calcularArea());
    }
}
```