Pesquisa POO (Projeto 3)

Pacotes

Um pacote é um grupo de módulos ou bibliotecas agrupadas para facilitar a organização, reutilização e distribuição do código da linguagem de programação. Isso permite que os desenvolvedores empacotem e compartilhem recursos de software com eficiência. Um pacote pode conter um conjunto de classes, funções, variáveis e outros recursos relacionados, permitindo que os desenvolvedores usem esses recursos em seus projetos sem precisar reescrever todo o código.

Segue um exemplo de código:

```
import java.lang.Math;


public class ExemploPacotes {
   public static void main(String[] args) {
      // Utilizando o método sqrt() da classe Math para calcular a raiz quadrada
      double num = 16;
      double raizQuadrada = Math.sqrt(num);
      System.out.println("A raiz quadrada de " + num + " é " + raizQuadrada);
   }
}
```

Classes Abstratas

Uma classe abstrata em Java é uma classe que não pode ser instanciada diretamente, mas serve como base para outras classes derivadas. Ela pode conter métodos abstratos, que são declarados, mas não têm uma implementação concreta na classe abstrata. As classes derivadas são obrigadas a implementar os métodos abstratos definidos na classe abstrata. As classes abstratas são usadas para definir uma estrutura comum e compartilhar comportamentos entre várias classes relacionadas.

Segue um exemplo de classe abstrata:

```
abstract class Animal {
   public abstract void emitirSom();
```

```
    public void dormir() {
        System.out.println("Zzzzzz...");
    }
}
class Cachorro extends Animal {
    public void emitirSom() {
        System.out.println("Au au!");
    }
}

public class ExemploClassesAbstratas {
    public static void main(String[] args) {
        Cachorro cachorro = new Cachorro();
        cachorro.emitirSom();
        cachorro.dormir();
    }
}
```

Interface

Uma interface em Java é uma coleção de métodos abstratos (sem implementação) e constantes que define um contrato para uma classe implementar. Ela define a assinatura dos métodos que devem ser implementados pela classe que a utiliza. As interfaces permitem a implementação de múltiplas interfaces por uma classe, possibilitando a herança múltipla de comportamentos. Elas são usadas para definir comportamentos comuns que podem ser compartilhados entre diferentes classes, promovendo o polimorfismo e a modularidade do código.

Segue o exemplo de interface:

```
interface Veiculo {
    void acelerar(int velocidade);
    void frear();
```

```
}
class Carro implements Veiculo {
   public void acelerar(int velocidade) {
      System.out.println("O carro está acelerando a " + velocidade + " km/h");
   }
   public void frear() {
      System.out.println("O carro está freando");
   }
}
public class ExemploInterfaces {
   public static void main(String[] args) {
      Carro carro = new Carro();
      carro.acelerar(100);
      carro.frear();
   }
}
```

Polimorfismo

Polimorfismo é um conceito importante na programação orientada a objetos que permite que objetos de diferentes classes sejam tratados de forma genérica, por meio de uma referência de tipo mais abstrato. Isso significa que um objeto de uma classe derivada pode ser atribuído a uma variável de tipo da classe base, permitindo que métodos sejam chamados de forma polimórfica, isto é, a implementação adequada do método é determinada em tempo de execução com base no tipo real do objeto.

Segue o exemplo em código:

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

```
class Cachorro extends Animal {
    public void emitirSom() {
        System.out.println("O cachorro late: Au au!");
    }
}
class Gato extends Animal {
    public void emitirSom() {
        System.out.println("O gato mia: Miau!");
    }
}
public class ExemploPolimorfismo {
    public static void main(String[] args) {
        Animal animal1 = new Cachorro();
        Animal animal2 = new Gato();
        animal1.emitirSom(); // Chama o método emitirSom() da classe Cachorro
        animal2.emitirSom(); // Chama o método emitirSom() da classe Gato
    }
}
```