Atividade de Programação Orientada a Objetos (POO)

Título: Modelando o Mundo com Classes: Uma Colaboração entre Aluno e IA

🎯 Objetivo da Atividade

Com o auxílio de uma ferramenta de Inteligência Artificial (IA) generativa (como Gemini, ChatGPT, ou similar), o aluno deverá **criar um programa simples em Java** que demonstre, de forma prática, a aplicação dos quatro pilares fundamentais da Programação Orientada a Objetos (POO).

O foco é usar a IA não apenas para gerar código, mas como um **tutor e auxiliar** na estruturação do projeto, garantindo que os conceitos de POO sejam corretamente aplicados.

🛠️ Ferramentas e Requisitos

1. **Linguagem de Programação:** Java (Versão 8 ou superior).
2. **Ambiente de Programação:** JDK, IDE (IntelliJ, Eclipse) ou editor de texto com terminal (VS Code).
3. **Ferramenta de IA:** Acesso a uma IA generativa de texto (conexão com a internet).

📝 Roteiro de Trabalho: A Missão POO

Fase 1: Análise e Abstração (O que modelar?)

1. **Escolha do Tema:** Escolha um domínio simples para modelar (ex: Animais, Veículos, Pessoas, Produtos).
2. **Criação da Classe Base (Superclasse):** Use a IA para criar uma Superclasse que represente a essência do seu domínio.
    - **Conceito principal abordado: Abstração** (focar no essencial).
    - *Exemplo de Prompt:* "Crie a classe base Animal em Java com atributos essenciais (nome, especie) e um método de ação (emitirSom())."

Fase 2: Implementação e Controle (Encapsulamento e Herança)

1. **Proteção de Dados (Encapsulamento):** Peça à IA para modificar a classe base para "proteger" um atributo (ex: idade ou saldo) tornando-o **privado** (private). Adicione métodos de acesso controlados **públicos** (get e set) a este atributo.
    - **Conceito principal abordado: Encapsulamento** (proteção e controle de dados).
2. **Criação das Subclasses (Herança):** Crie duas subclasses (Classes Filhas) que **herdam** (extends) da Superclasse.

- **Conceito principal abordado: Herança** (reaproveitamento de código).

Fase 3: Muitas Formas (Polimorfismo)

1. **Sobrescrita de Métodos:** Garanta que cada subclasse sobrescreva (redefina) o método de ação da classe base (ex: o método emitirSom() de Animal é diferente em Cachorro e Gato). Use a anotação @Override.
   - **Conceito principal abordado: Polimorfismo** (o mesmo método se comporta de formas diferentes em objetos distintos).
2. **Teste:** Crie instâncias de todas as classes e chame o método polimórfico para demonstrar a diferença de comportamento.

Fase 4: Documentação para a Apresentação em Sala

Preencha a Tabela de Análise de Código (abaixo) com o código final e as explicações necessárias.

📚 Tabela de Análise de Código (Para Preenchimento)

Conceito da POO	Linhas do Código que o Demonstram	Explicação (Como o código aplica o pilar)
Abstração		
Encapsulamento		
Herança		
Polimorfismo		

🗣️ Apresentação e Discussão em Sala

Cada aluno ou grupo deverá apresentar o código final e responder às seguintes perguntas:

1. Qual foi o tema escolhido para a modelagem?
2. Onde no código está definida a **Abstração** do seu objeto do mundo real?
3. Como você demonstrou o **Encapsulamento**? Por que é importante proteger esse atributo?
4. Quais classes demonstram **Herança** e como isso otimiza o código?
5. Qual método exemplifica o **Polimorfismo** e quais são as "muitas formas" que ele assume no seu programa?