Universidade de Brasília

Faculdade UnB Gama

UNIVERSIDADE DE BRASÍLIA - UNB
ENGENHARIA DE SOFTWARE - GRADUAÇÃO

CAIO VITOR CARNEIRO DE OLIVEIRA - 200057227
LUCAS LOPES ROCHA - 202023903

TRABALHO 3

RELATÓRIO DAS PERGUNTAS

Brasília
2024

1. Definição dos Princípios de Bom Projeto de Código e Relacionamento com Maus-Cheiros

Simplicidade:
Um código simples é direto e sem complexidade desnecessária. Ele faz apenas o que precisa ser feito, sem incluir funcionalidades extras ou detalhes desnecessários.

- **Maus-cheiros relacionados:** *Código Duplicado*, *Código Morto*, *Complexidade desnecessária* .

Elegância:
Um código elegante é agradável de ler e usa soluções claras e concisas para problemas complexos.

- **Maus-cheiros relacionados:** *Soluções desajeitadas*, *Métodos Longos*.

Modularidade:
Refere-se à separação do código em partes menores e independentes que podem ser desenvolvidas, testadas e mantidas de forma isolada.

- **Maus-cheiros relacionados:** *Classes Grandes*, *Métodos Longos*, *Código Duplicado*.

Boas Interfaces:
Interfaces bem definidas facilitam a interação entre diferentes partes do sistema e garantem que as partes possam evoluir independentemente.

- **Maus-cheiros relacionados:** *Intimidade Inapropriada*, *Mudança Divergente*.

Extensibilidade:
Um sistema extensível pode ser facilmente modificado para incorporar novas funcionalidades sem a necessidade de alterações profundas.

- **Maus-cheiros relacionados:** *Rigidez*, *Imobilidade*.

Evitar Duplicação:
A duplicação de código aumenta a complexidade e dificulta a manutenção.

- **Maus-cheiros relacionados:** *Código Duplicado*, *Código Morto*.

Portabilidade:
Um código portátil pode ser executado em diferentes ambientes com o mínimo de adaptação.

- **Maus-cheiros relacionados:** *Dependências*, *Código de Plataforma Específica*.

Código Idiomático e Bem Documentado:
Um código que segue as convenções da linguagem e é bem documentado é mais fácil de entender e manter.

- **Maus-cheiros relacionados:** *Nomes Ruins*, *Comentário Exagerado*.

2. Maus-Cheiros de Código e Princípios Violados

- **Código Duplicado:**
  - *Exemplo:* O cálculo de impostos está presente tanto na classe `CalculoImpostos` quanto no método `calcularImpostos()` da classe `Venda`.
  - **Princípios Violados:** *Simplicidade* e *Evitar Duplicação*.
  - **Refatoração Sugerida:** *Extrair Método* - Unificar a lógica de cálculo de impostos em um único método na classe `CalculoImpostos` e chamá-lo a partir de `Venda`.

- **Métodos Longos:**
  - *Exemplo:* O método `calcularFrete()` na classe `Venda` contém uma lógica complexa com muitos ramos de decisão, o que torna o método longo e difícil de manter.
  - **Princípios Violados:** *Simplicidade* e *Modularidade*.
  - **Refatoração Sugerida:** *Extrair Método* - Quebrar o método em métodos menores, cada um responsável por calcular o frete para um grupo de estados.

- **Responsabilidade Múltipla:**
  - *Exemplo:* A classe `Venda` lida com a adição de produtos, cálculo de valor total, cálculo de frete e impostos, o que sugere uma sobrecarga de responsabilidades.
  - **Princípios Violados:** *Modularidade* e *Boas Interfaces*.
  - **Refatoração Sugerida:** *Extrair Classe* - Mover a lógica de cálculo de frete para uma classe específica, como `CalculoFrete`, e o cálculo de

impostos para `CalculoImpostos`.

- **Intimidade Inapropriada:**
  - *Exemplo:* O método `calcularValorTotal()` da classe `Venda` manipula diretamente a lista de produtos da classe `CalculoVenda`.
  - **Princípios Violados:** *Boas Interfaces*.
  - **Refatoração Sugerida:** *Introduzir Método na Classe Colaboradora* - Em vez de manipular diretamente a lista, criar métodos na classe `CalculoVenda` para adicionar produtos e calcular o valor total.

- **Switch Statements:**
  - *Exemplo:* O uso de `switch` para calcular frete baseado no estado na classe `Venda`.
  - **Princípios Violados:** *Extensibilidade*.
  - **Refatoração Sugerida:** *Substituir Condicional por Polimorfismo* - Criar uma hierarquia de classes ou usar um mapa para encapsular a lógica de cálculo de frete por estado.