Exercícios

1) Quais são as vantagens e as desvantagens de usar padrões de projetos no desenvolvimento de software?

   R: Pode-se citar as vantagens de aumento de produtividade; diminuição do tempo de desenvolvimento; qualidade do código e do tratamento de dados; flexibilidade na estrutura do software; manutenibilidade; reusabilidade; tratamento de erros comuns do desenvolvimento.

2) Pesquise o que são padrões GRASP e encontre três exemplos.

   R: Os padrões GRASP englobam uma série de princípios baseados em conceitos de Orientação Objetos. Partindo de análises que procuram definir quais as obrigações dos diferentes tipos de objetos em uma aplicação, estes patterns disponibilizam uma conglomerado de recomendações que têm por função favorecer a obtenção de sistemas melhor estruturados.

   Alguns exemplos de padrões GRASP são:

   - **Controller: Diminui o acoplamento entre os dados e sua representação;**
   - **Creator: Responsável por criar um objeto;**
   - **Information Expert: A classe que tem a informação necessária para satisfazer a responsabilidade.**

3) Escolha um padrão GoF e implemente um pequeno exemplo em sua linguagem de programação favorita.

   R: Código:

```
from abc import ABC, abstractmethod

# 1. Criando a interface de conversores (AbstractProduct)
class Conversor(ABC):
    @abstractmethod
    def converter(self, numero: int) -> str:
        pass

# 2. Implementando o conversor de decimal para binário (ConcreteProduct)
class ConversorDecimalBinario(Conversor):
    def converter(self, numero: int) -> str:
        return bin(numero)[2:]

# 3. Implementando a Abstract Factory
```

```
class ConversorFactory(ABC):
    @abstractmethod
    def criar_conversor(self) -> Conversor:
        pass

# 4. Fábrica concreta que cria o conversor de decimal para binário (ConcreteFactory)
class ConversorDecimalBinarioFactory(ConversorFactory):
    def criar_conversor(self) -> Conversor:
        return ConversorDecimalBinario()

# 5. Cliente
def cliente(fabrica: ConversorFactory, numero: int) -> str:
    conversor = fabrica.criar_conversor()
    return conversor.converter(numero)

# Usando a Abstract Factory
numero_decimal = 42
fabrica_binario = ConversorDecimalBinarioFactory()
resultado_binario = cliente(fabrica_binario, numero_decimal)

print(f"O número decimal {numero_decimal} em binário é: {resultado_binario}")
```

4) Pesquise o que são padrões GoF e encontre cinco exemplos.

R: Os padrões GoF (Gang of Four) são um conjunto de 23 padrões de projetos catalogados no livro "Design Patterns: Elements of Reusable Object-Oriented Software", escrito por Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides, quatro autores (daí o apelido Gang of Four).

Os padrões GoF são categorizados em três grupos principais:

- **Padrões de Criação: Tratam do processo de criação de objetos, fornecendo maneiras de instanciar objetos de forma mais flexível e controlada (Abstract Factory);**
- **Padrões Estruturais: Lidam com a composição de classes e objetos para formar estruturas maiores e mais complexas (Proxy); e**
- **Padrões Comportamentais: Se concentram nas maneiras como classes e objetos interagem e como responsabilidades são distribuídas**

(Template Method).

5) Pesquise e descubra o nome do padrão de projeto adotado na atividade prática realizada nesta aula.

   R: O nome do padrão de projeto utilizado na atividade prática se chama Abstract Factory. Esse padrão pertence à categoria de Projetos de Criação do GoF. É evidente que é essa, pois existe uma classe abstrata que fabrica um modelo padrão para criação de subclasses dentro do código.