INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO MARANHÃO
CAMPUS SÃO LUÍS – MONTE CASTELO
DIRETORIA DE ENSINO SUPERIOR
DEPARTAMENTO DE COMPUTAÇÃO

Curso de Sistemas de Informação

Disciplina: Padrões de Software
Profº João Carlos Pinheiro

Data de Entrega: 29/04/2022

Etapa 01 – Atividade 02

1. Qual é propósito de estudar Padrões de Projeto?

2. Descreva os quatro elementos essenciais de um padrão de projeto?

3. Explique como padrões de projeto podem ser utilizados especificamente para projetar frameworks.

4. Comente a afirmação: "Padrões de Projeto criam um vocabulário comum de projeto".

5. Comente a afirmação: "Padrões de Projeto ajudam a documentar o projeto".

6. Se eu tenho apenas uma única estratégia, faz sentido implementar o Strategy?

7. Crie todo o mecanismo para flexibilizar a criação de diferentes estratégias de impostos. Crie a interface Imposto, e as estratégias ICMS e ISS. O ISS deve ser 6% do valor do orçamento, e o ICMS deve ser 25% do valor do orçamento. Crie a classe Orcamento, que tem como atributo um valor. Crie um construtor que recebe esse valor, e um getter para devolvê-lo. Crie a classe CalculadorDeImposto, que recebe um Orcamento e um Imposto. Essa classe calcula o imposto usando a estratégia recebida e imprime o resultado na tela.

8. Uma livraria vende diversos artigos consistindo de Livros, Revistas e Jogos.

   Há livros e revistas impressos e digitais.

   A livraria possui Jogos de Tabuleiro e Jogos de Vídeo Game, sendo o último o jogo físico ou digital.

   No pagamento a vista, temos a seguinte política de descontos:

   - Livros, Revistas Físicos e Jogos de Tabuleiro: 30% de desconto.
   - Livros e Revistas digitais: 15% de desconto.
   - Não há desconto para Jogos de Vídeo Game.

   Porém, a livraria pode ter promoções especiais durante o ano, de modo que os descontos dos produtos podem mudar.

   Implemente sua solução.

9. Em quais situações se torna ideal o uso do padrão Chain of Responsability? Cite algum momento que poderia ter aplicado ele na sua experiência até hoje

10. Utilizando o padrão Chain of Responsability, crie um caixa eletrônico capaz manipular requisições de saque corretamente, sempre considerando *o menor número de notas possível*.

    Por exemplo, em uma solicitação de saque no valor de R$475, o caixa deve entregar:

    - 4 notas de R$100
    - 1 nota de R$50
    - 1 nota de R$20
    - 1 nota de R$5

    Para isto, crie manipuladores para contar a quantidade de cada tipo de nota. Ao terminar sua contagem, cada manipulador deve passar ao próximo o cálculo da quantidade de notas relativas ao montante restante.

    Cada manipulador deve exibir a sua contagem.

11. Um servidor de aplicação bancária que se comunica com outros deve responder de várias formas diferentes, de acordo com a solicitação da aplicação cliente.

    Se a aplicação solicitar uma **`Conta`**, cujos atributos formatados em JSON, por exemplo, o servidor deverá responder nesse formato; se a aplicação solicitar XML, o servidor deverá responder XML; se ela pedir no formato CSV, separado por % (por cento), a aplicação deverá devolver dessa forma.

Implemente um `Chain of Responsibility` onde, dada uma requisição e uma conta bancária, ela passeia por toda a corrente até encontrar a classe que deve processar a requisição de acordo com o `formato` solicitado, e imprime na tela a conta bancária no formato correto.

Imagine que a classe `Requisição` possui um *getter* `getFormato()`, que responde "XML", "CSV", ou "PORCENTO", indicando qual tratamento adequado. Uma `Conta` possui apenas saldo e nome do titular:

```
1 enum Formato {
2   JSON,
3   XML,
4   CSV
5 }
6
7 class Requisicao {
8   private Formato formato;
9   public Requisicao(Formato formato) {
10    this.formato = formato;
11  }
12
13  // getter para o Formato
14}
```

{ numero: 1234, saldo:2300 }

<conta > <numero>1234</numero> <saldo> 2300</saldo></conta>

1234%2300