UNIVALI POLITECNICA KOBRASOL Ciência da Computação Programação 3ºper
TRABALHO M3 (25/1) – Implementação OO utilizando Padrões de Projeto

Escolher um problema qualquer, descrevê-lo (objetivo, elementos, funcionalidades), gerar a UML (diagrama de classes) e desenvolver um sistema orientado a objetos em Java utilizando 2 padrões de projeto GoF/MVC, dentre os indicados abaixo. Pode-se remodelar os trabalhos da M1 e M2 aplicando padrões sobre eles e utilizá-los como problema.

O sistema deverá ter no mínimo 3 classes de domínio e 2 relacionamentos.

Padrões de projeto apresentados:

- Arquitetura MVC (*model-view-controller*): padrão em *design* de software muito usado para implementar interfaces de usuário, dados e lógica de controle. Ele enfatiza a separação entre a lógica de negócios e a exibição do software.
- Padrão Singleton: Assegura que uma classe tenha apenas uma instância e provê um ponto global de acesso a ela.
- Padrão Iterator: Provê um modo de acesso a elementos de um agregado de objetos, sequencialmente, sem exposição de estruturas internas.
- Padrão Adapter: Converte a interface de uma classe em outra, esperada pelo cliente. Permite que classes que antes não poderiam trabalhar juntas, por incompatibilidade de interfaces, possam agora fazê-lo.
- Padrão Builder: Provê uma interface genérica para a construção incremental de agregações. Um Builder esconde os detalhes de como os componentes são criados, representados e compostos.
- Padrão Decorator: Anexa responsabilidades adicionais a um objeto dinamicamente. Provê uma alternativa flexível para extensão de funcionalidade, sem ter que usar Herança.
- Padrão Facade: Provê uma interface unificada para um conjunto de interfaces em um subsistema. Define uma interface de alto nível para facilitar o uso deste subsistema.
- Padrão Factory Method: Define uma interface para criação de um objeto, permitindo que as suas subclasses decidam qual classe instanciar – deixa a responsabilidade de instanciação para as subclasses.
- Padrão Memento: Captura e externaliza o estado interno de um objeto (captura um "snapshot"). Não viola o encapsulamento.
- Padrão Observer: Provê sincronização, coordenação e consistência entre objetos relacionados.
- Padrão Prototype: Especifica os tipos de objetos a serem criados num sistema, usando uma instância protótipo. Cria novos objetos copiando este protótipo.
- Padrão State: Deixa um objeto mudar seu comportamento quando seu estado interno muda, mudando, efetivamente, a classe do objeto.
- Padrão Strategy: Define uma família de algoritmos, encapsula cada um deles, e torna a escolha de qual usar flexível. Desacopla os algoritmos dos clientes que os usa.

Lembrete: o material de todos os seminários apresentados foi disponibilizado no AVA.

Avaliação

Os requisitos do trabalho devem ser entregues de forma consistente, implementando os conceitos de POO e aplicando os padrões de projeto GoF/MVC corretamente.

Trabalhos com erros de compilação ou que não atendam os requisitos receberão nota zero.

Pontuação	2,0	2,0	6,0
Item	Descrição+UML	Aplicação dos padrões	Implementação

O trabalho será desenvolvido em trio, em linguagem Java, postado via arquivo compactado no link da atividade **até 19h de 10/julho/25** (5af). Os nomes dos alunos do grupo devem constar do nome do arquivo (ex. Joao_PedroT3POO.rar). Nesta mesma data, a partir das 19h haverá defesa do projeto (10-15min) e o **TRABALHO TAMBÉM DEVERÁ SER ENTREGUE IMPRESSO (descrição, UML e código).**

Para efeito de nota: a postagem do trabalho só terá validade após realização da defesa na data acima.