ESPECIFICAÇÃO DE TRABALHO PRÁTICO FINAL

Análise e Desenvolvimento de Sistemas
INF008 – Programação Orientada a Objetos
Professor: Sandro Santos Andrade

1. Objetivo Geral

Este exame prático foi elaborado para avaliar seu entendimento e aplicação dos principais conceitos de programação orientada a objetos, incluindo tópicos avançados como **reflexão** e **gerenciamento de plugins em tempo de execução**. Você deverá projetar, implementar e documentar um **Sistema de Gerenciamento de Biblioteca (SGB)** que suporte variação de funcionalidades por meio de plugins. O sistema deve demonstrar seu domínio dos seguintes conceitos de POO:

- Encapsulamento
- Herança
- Polimorfismo
- Abstração
- Associação, Agregação e Composição ▪ O trabalho deve ser enviado até o dia 08/12/2024.
- ▪ O código-fonte e o arquivo README.md deve ser enviado por email para sandroandrade@ifba.edu.br, com o assunto "INF008 T1 <seu-nome-completo>", sem as aspas.
- ▪ Trabalhos enviados com outro assunto não serão corrigidos
- Tratamento de Exceções
- Reflexão
- Arquitetura de Plugins
- Persistência de Dados
- Reusabilidade e Manutenibilidade do Código

2. Requisitos Funcionais

Você deve projetar e implementar um **Sistema de Gerenciamento de Biblioteca (SGB)** que suporte **gerenciamento de plugins em tempo de execução**. O sistema deve gerenciar livros, membros da

biblioteca e transações de empréstimo, além de permitir que plugins estendam sua funcionalidade de forma dinâmica. Os plugins podem ser adicionados, removidos e selecionados em tempo de execução, e o sistema deve usar **reflexão** para carregar e interagir com esses plugins. Além disso, o sistema deve implementar **persistência de dados**, salvando o estado atual em um arquivo binário ao sair e carregando os dados ao iniciar.

Requisitos funcionais:

- Gerenciamento de Livros:
  - Cada livro possui um ISBN único, título, autor, ano de publicação e gênero.
  - Assuma que cada livro possui apenas uma cópia na biblioteca.
  - O sistema deve permitir o cadastro de livros.
- Gerenciamento de Usuários:
  - Cada usuário possui um ID único, nome e uma lista de livros emprestados.
  - O sistema deve permitir a criação de usuãrios.
  - Usuários podem emprestar e devolver livros (ver "transações de empréstimo").
- Transações de Empréstimo:
  - Informar o usuário que está realizando o empréstimo.
  - Livros podem ser pesquisados por título (mostrar apenas livros disponíveis, ou seja, não emprestados).
  - Um usuário pode tomar emprestado até 5 livros por vez.
  - A data em que o empréstimo ocorreu deve ser informada no momento do empréstimo (pode ser uma data retroativa).
  - Cada livro pode ser emprestado por um máximo de 14 dias.
- Relatórios (plug-ins):
  - Gerar um relatório de todos os livros emprestados no momento.
  - Gerar um relatório de todos os livros atrasados e as multas correspondentes.
    - O sistema deve rastrear datas de vencimento e calcular multas para livros atrasados (taxa de multa: R$ 0,50 por dia por livro).
  - Os relatórios devem ser implementados como plug-ins.
- Persistência de Dados:
  - Ao sair, o sistema deve salvar o estado atual (livros, membros, transações) em um arquivo binário.
  - Ao iniciar, o sistema deve carregar os dados do arquivo binário, restaurando o estado anterior.
- Não é necessário implementar autenticação e autorização no sistema.
- O sistema deverá possuir interface gráfica de usuário.

3. Requisitos Técnicos

- Qualidade do Código:
  - Use nomes significativos para variáveis e métodos.
  - Siga padrões e convenções de codificação adequados.
  - Garanta que o código esteja bem documentado com comentários onde necessário.
- Design:
  - Garanta que o sistema seja modular e extensível.
- Tratamento de Erros:
  - Implemente tratamento de exceções para gerenciar entradas inválidas e casos extremos.
- Reflexão:
  - Use reflexão para carregar e interagir com plug-ins dinamicamente.
- Persistência:
  - Implemente a funcionalidade de salvar e carregar dados em um arquivo binário de forma eficiente.

4. Diretrizes de Implementação

- Design das Classes:
  - Projete classes que representem as entidades do sistema (por exemplo, Book, User, Loan).
  - Use herança, interfaces, polimorfismo e ligação dinâmica quando apropriado.
- Estruturas de Dados:
  - Use estruturas de dados apropriadas (por exemplo, listas, mapas) para armazenar livros, usuários e transações.
- Interface de Usuário:
  - O sistema deve apresentar uma Interface Gráfica de Usuário (GUI) simples e intuitiva.
  - A GUI deve permitir que os usuários realizem todas as operações necessárias (por exemplo, adicionar um livro, emprestar um livro e visualizar relatórios).
- Persistência de Dados:
  - Implemente a funcionalidade de salvar o estado do sistema em um arquivo binário ao sair.
  - Implemente a funcionalidade de carregar o estado do sistema a partir de um arquivo binário ao iniciar.
- Testes:

- Garanta que seu código esteja livre de bugs e lide com casos extremos de forma adequada.

5. Entregáveis

- Código-Fonte:
  - Envie todos os arquivos de código-fonte, incluindo arquivos de teste.
  - Garanta que seu código esteja bem organizado e siga a estrutura de diretórios apropriada para a linguagem escolhida.
- Documentação:
  - Forneça um arquivo README que explique como compilar e executar seu programa.

6. Critérios de Avaliação

Critério: Correção (5,0 pontos) - O sistema deve implementar corretamente todos os requisitos funcionais.

Critério: Qualidade do Código (3,0 pontos) - O código deve estar limpo, bem organizado e seguir as melhores práticas.

Critério: Design e Arquitetura (1,0 ponto) - Uso adequado dos princípios de POO, modularidade e extensibilidade.

Critério: Persistência de Dados (1,0 ponto) - Implementação correta da funcionalidade de salvar e carregar dados.

7. Prazo e Forma de Entrega

- O trabalho deve ser enviado até o dia **25/02/2025**.
- O código-fonte e o arquivo README.md deve ser enviado por email para sandroandrade@ifba.edu.br, com o assunto "INF008 T2 <seu-nome-completo>", sem as aspas.
- Trabalhos enviados com outro assunto não serão corrigidos

8. Informações Importantes

- Todos os códigos-fonte entregues serão checados por plágio utilizando as ferramentas **MOSS** (Measure of Software Similarity - https://theory.stanford.edu/~aiken/moss/) e **JPlag** (https://github.com/jplag/JPlag). Desenvolva sua própria solução, é sua chance de aprender.
- Haverá um avaliação subsequente para elucidação de dúvidas sobre o código-fonte que você desenvolveu. O formato e data serão ainda definidos.
- Todas as dúvidas sobre o trabalho deverão ser abertamente discutidas no grupo da disciplina no Telegram.

Bom trabalho!