Universidade Federal do Rio Grande do Sul
Departamento de Informática Aplicada

INF01057 - Programação Orientada a Objeto
2024/2

Trabalho Final

Leia atentamente este documento. Nele estão descritos todos os requisitos mínimos da aplicação a ser implementada, entregas e os critérios de avaliação.

Objetivo

O Trabalho Final objetiva exercitar as habilidades e conceitos de programação desenvolvidos ao longo do semestre através da implementação de uma aplicação orientada a objetos utilizando a linguagem de programação Java.

Orientações

Os alunos poderão realizar os trabalhos de forma individual ou formar grupos de dois ou três integrantes e definir que aplicação desejam implementar como trabalho final. Isto é, a escolha do tema/problema a ser abordado através da aplicação a ser implementada é livre para cada grupo**.** Vocês podem pensar em implementar um jogo, um sistema que esteja alinhado com alguma necessidade que vocês observam dentro da Universidade, da suas áreas de atuação ou de suas experiências profissionais. Sejam criativos! Além da definição do problema, os alunos serão responsáveis pela interpretação, modelagem da solução e implementação da aplicação, de forma que estejam de acordo com os princípios de orientação a objetos vistos na disciplina.

Requisitos

- A implementação da aplicação deve ser realizada em Java. O grupo pode utilizar a IDE de sua preferência, desde que, ao final, sejam fornecidos corretamente os arquivos solicitados (vide item “Entrega”). A implementação da aplicação precisa adotar técnicas de orientação a objetos aprendidas em aula de forma correta; i.e., não basta que a aplicação funcione corretamente mas sim que também aplique corretamente os pilares de POO
- A implementação deverá aplicar, obrigatoriamente, os conceitos de encapsulamento, classes, objetos, métodos, atributos, mensagens, herança, polimorfismo e classes abstratas e/ou interfaces. Todos estes conceitos devem estar representados no modelo e código entregues.
- A aplicação deve envolver entidades distintas, entre classes concretas e abstratas e interfaces. A quantidade fica a critério dos alunos, porém,

espera-se uma complexidade razolavel (por exemplo, dez entidades diferentes).

- Devem ser utilizados corretamente os modificadores de acesso a fim de garantir o encapsulamento. São obrigatórios a definição e uso de pacotes.
- A aplicação deve incluir uma interface de comunicação com o usuário (gráfica ou não) que o oriente quanto ao uso da aplicação e permita realizar as operações disponíveis e visualizar informações, de acordo com o problema abordado. Atenção: não é necessário sofisticação em sua interface, o que será valorizado é a disponibilidade das operações, a apresentação das informações e a usabilidade da aplicação.
- Ao implementar a aplicação, deve-se preocupar com a consistência e coerência dos dados e das operações realizadas. Por exemplo, se a aplicação visar o gerenciamento de uma loja *online*, então só se deve efetuar a venda de itens com estoque disponível.
- A aplicação deve estar devidamente documentada. Isto inclui o uso apropriado de comentários no código, a descrição das responsabilidades das classes e interfaces e a escrita de um diagrama de classes (Perspectiva de Especificação) que represente a estrutura final do sistema (classes/interfaces e relacionamentos).

Desejáveis

- Criar e disponibilizar um arquivo executável **.jar** gerado a partir dos códigos (o **.jar** deve ser entregue juntamente com os códigos originais).
- Criar uma interface gráfica para auxiliar o uso da aplicação. Atenção: o uso de uma interface interativa (gráfica ou não) para guiar o usuário é obrigatória.
- Simular um “banco de dados” via manipulação de arquivos, de tal forma que se possa carregar/registrar informações relevantes da aplicação de forma não-volátil.
- Identificar possíveis exceções e propor formas de tratamento adequadas.

Entrega

O trabalho será entregue em 3 (três) etapas com datas definidas e sem extensão:

1. Sinalização e entrega da descrição do problema a ser modelado: até dia **28 de Novembro de 2024** às 23:59 horas através do e-mail **mffranco@inf.ufrgs**.
    - O grupo deverá adicionar no fórum “Grupos – Trabalho Final”, disponível no **Moodle** da disciplina, uma mensagem para a turma com as seguintes informações:
        - **Nome** dos(as) Integrantes (Máximo de três pessoas)
        - **Descrição** do problema e da aplicação que será modelada (mínimo de 1-2 parágrafos)

2. Diagrama de Classes deverá ser enviado até o dia **10 de Dezembro**: Deverão constar as entidades abstraídas (classes e interfaces), bem como lista preliminar de atributos de cada classe a partir do contexto (refinamentos podem ser feitos em momento posterior). Se já houverem definições a respeito de funcionalidades e relacionamentos entre classes, estes também podem constar no documento.

- O relatório será analisado pelo professor e, se necessário, discutido com o grupo a fim de deixá-lo mais alinhado com as especificações dos requisitos mínimos. Detalhes sobre classes e funcionalidades podem ser mudados durante a implementação, mas os alunos deverão manter até o final do semestre a proposta de projeto (tema/aplicação definido(a)).

3. Entrega do código: até dia **10 de Janeiro de 2025** às 23:59 horas pelo ambiente virtual Moodle.

- *Upload* no ambiente Moodle de um único arquivo compactado (**.zip**). Este arquivo deve conter o código completo e documentado (arquivos .java são obrigatórios), bem como o diagrama de classes final em Perspectiva de Especificação (formato imagem ou **PDF**). O arquivo **.jar**, se gerado, deve ser incluído neste mesmo arquivo compactado.
- Descrevam o nome dos autores no cabeçalho do arquivo que contém o método principal, em forma de comentário.
- Vocês podem fazer *upload* de diferentes versões da aplicação nesse prazo. Identifiquem adequadamente as versões, tal que a mais recente seja facilmente identificada. Façam o *upload* assim que tiver uma versão funcional, de modo a garantir a entrega e precaver-se de problemas com servidor, rede, etc.

4. Apresentação síncrona: dia **14 de Janeiro de 2025** das 08:30 às 10:10 *em sala de aula*.

- Cada grupo deve apresentar (a) o problema, (b) a modelagem e (c) as principais funcionalidades da aplicação desenvolvida. Tais funcionalidades também devem ser exibidas na execução da aplicação. É obrigatório que todos os alunos do grupo expliquem, ao menos, um dos pontos ((a), (b) ou (c)).
- A apresentação, por grupo, deve ter no máximo 10 minutos de duração. A elaboração de uma “apresentação” objetiva e clara, que se adeque a este tempo, faz parte do processo de avaliação.
- Após a data de entrega do código, nenhuma alteração referente ao desenvolvimento da aplicação será permitida.
- A apresentação do trabalho conta nota na avaliação final.
- Caso não seja possível que todos os grupos apresentem durante o tempo de aula para a turma, será agendado horário diretamente com o professor para apresentações individuais durante a semana.

Avaliação

A avaliação deste Trabalho Final possui um peso de **25%** no cálculo da média final do aluno. Lembre-se que a não entrega desta atividade **implica em Recuperação automática**. A aplicação modelada/desenvolvida deve atender a todos os requisitos listados neste enunciado, deve obrigatoriamente estar orientada a objetos, não deve apresentar erros de compilação e deve executar normalmente. A aplicação desenvolvida deverá demonstrar os seguintes conteúdos que serão avaliados:

1. Habilidade em estruturar aplicações pela decomposição da realidade modelada em classes/interfaces. Cada classe deve ser responsável pelo seu estado interno (atributos) e propor uma série de funcionalidades para alterar o estado interno (métodos). **(3 pontos)**
2. Documentação de programas (indentação, utilização de nomes de variáveis/constantes adequados, abstração dos procedimentos para obter maior clareza, uso de comentários no código, etc.). (**1,5 ponto)**
3. Domínio na utilização dos conceitos estudados: encapsulamento, herança, polimorfismo, classes abstratas, interface, sobrecarga (*overload*) e sobreescrita (*override*) de métodos, atributos de instância ou de classe, métodos de instância ou de classe e modificadores de acesso. (**3 pontos)**
4. Formatação e controle de entrada e saída, com construção de interfaces que orientem corretamente o usuário, sem instruções ou intervenção adicional do programador. Garantia da consistência e coerência dos dados e operações. **(1,5 pontos)**
5. Clareza, organização e demonstração de domínio da solução implementada serão avaliados na apresentação. **(1 ponto)**

Observações

Este trabalho deve refletir a solução individual do grupo para o problema proposto. Casos de plágio serão tratados com severidade e resultarão em anulação da nota e, consequentemente, na reprovação dos alunos envolvidos (vide programa da disciplina). Para detectar e quantificar o plágio no código, será utilizado o *software* MOSS.

Dicas

- Para auxiliar na elaboração da interface gráfica, caso o grupo deseje implementar uma, sugiro adotar uma IDE. Explorem a documentação das bibliotecas gráficas *Swing* (recomendada) ou AWT. O NetBeans, por exemplo, tem um tutorial ilustrando como se pode projetar interfaces gráficas com *Swing*. Tutoriais Oracle sobre *Swing* também são boas referências públicas.

- A apresentação “*Material Extra: exportando um arquivo .jar*”, disponibilizada no ambiente virtual Moodle, explica como exportar o arquivo .jar via linha de comandos.
- Faça bom proveito do *Google, ChatGPT* e da infinidade de informações que eles fornecem! Porém, saibam o que vocês estão desenvolvendo e tenham muito cuidado com utilizar códigos prontos sem compreende-lo. Grupos que não souberem explicar seu programa ou que tenham código amplamente gerado por IA poderão ser penalizados ou ter a nota anulada.