Trabalho Final

Estruturas de Dados I – 2023/2

Prof. Dr. Thiago L. T. da Silveira
tltsilveira@inf.ufrgs.br

Instruções

- Leia este documento **atentamente**. Nele, estão descritos os requisitos mínimos e desejáveis (extras) quanto à implementação do Trabalho Final. Também são apresentadas diretrizes para entrega e critérios de avaliação.

Enunciado Geral

Objetivo

O objetivo principal do Trabalho Final é exercitar algumas das *estruturas de dados* estudadas ao longo do semestre através da implementação de um **jogo de computador** utilizando a linguagem C. Objetivos secundários incluem exercitar as habilidades de pesquisa – uma vez que o aluno poderá utilizar bibliotecas e conceitos que não foram trabalhados em aula para implementar certas funcionalidades – e de trabalho em equipe – uma vez que este deve ser realizado *por duplas* ou *trios*. O Trabalho Final *não* pode ser implementado individualmente.

Orientações

Cada grupo deverá implementar um dos seguintes jogos (escolhido por sorteio):

- **Uno** (https://pt.wikipedia.org/wiki/Uno_(jogo_de_cartas)),
- **Rouba-monte** (https://pt.wikipedia.org/wiki/Rouba-monte) e
- **Desconfia** (https://pt.wikipedia.org/wiki/Desconfia).

Cada grupo é responsável pela modelagem (isto é, identificar formas de implementar cada regra e representação) e implementação do jogo. Atente que diferentes versões de cada jogo podem existir. É importante implementar uma delas de forma adequada. Deve-se utilizar estruturas de dados selecionadas e suas operações como *parte fundamental* da solução proposta – conforme discutido a seguir.

Requisitos Básicos

Todos os jogos devem seguir um conjunto mínimo de requisitos básicos:

Requisitos Básicos Gerais

- Deve-se usar os conceitos de tipos abstratos de dados (TADs). Tanto as estruturas de dados genéricas (que modelam coleções de dados) quanto outras entidades modeladas devem ser representadas por TADs. Estruture os TADs de forma adequada em arquivos de implementação e interface.
- Deve-se implementar uma interface (textual) intuitiva que apresenta um menu e o espaço de jogo. O menu deve ter, no mínimo, opções para iniciar um jogo, voltar ao jogo corrente e encerrar o jogo. Vocês poderão pesquisar a utilização de bibliotecas para criação de uma interface amigável para o jogo. Jogabilidade é fundamental. A escolha da biblioteca é responsabilidade dos grupos. Algumas bibliotecas sugeridas são a `conio2` e a `ncurses`, usadas para desenvolver interfaces em modo texto, para Windows e para Linux respectivamente.
- Deve-se documentar e modularizar adequadamente o código do jogo. Legibilidade também é fundamental.

Requisitos Básicos Individuais

Além dos requisitos básicos gerais, cada opção de jogo tem um conjunto específico de requisitos básicos:

Requisitos Básicos para a implementação do jogo Uno

- O monte de cartas (baralho) deve ser representado por uma pilha.
- A ordenação das jogadas deve ser representada por uma lista circular dupla.
- As cartas da mesa devem ser representadas por uma pilha.
- As cartas de cada jogador devem ser representada por uma lista (simples ou dupla).

Requisitos Básicos para a implementação do jogo Rouba-monte

- O monte de cartas (baralho) deve ser representado por uma pilha.
- A ordenação das jogadas deve ser representada por uma lista circular.
- Cada monte deve ser representado por uma pilha.
- As cartas de cada jogador devem ser representada por uma lista (simples ou dupla).

Requisitos Básicos para a implementação do jogo Desconfia

- O monte de cartas (baralho) deve ser representado por uma pilha.
- A ordenação das jogadas deve ser representada por uma lista circular.
- As cartas da mesa devem ser representadas por uma pilha.
- As cartas de cada jogador devem ser representada por uma lista (simples ou dupla).

Lembre-se de não infringir os protocolos associados a estruturas de dados como pilhas, filas e deques ao implementar novas operações (ex., embaralhamento). Os jogos podem ser implementados em formato *multiplayer* ou contra um ou mais jogadores. controlados pelo computador. Nesse caso, o computador deve realizar jogadas coerentes.

Extras

A pontuação associada aos extras pode extrapolar a nota 10 do Trabalho Final.

- Os grupos são encorajados a implementar estruturas de dados realmente genéricas (com uso de `void*` e *callbacks*). (0,85 ponto)
- Os grupos são encorajados a implementar seu jogo com uma interface gráfica (GUI) simples. Algumas bibliotecas que poderiam ser utilizadas são: a `Allegro`, a `raylib` e a `sdl2`. A `raylib` é altamente recomendada. A Figura 1 ilustra uma interface inicial em `raylib` para o jogo Uno. (1,15 ponto)
- Os grupos são encorajados a implementar efeitos sonoros em diferentes ações do jogo. (0,1 ponto)
- Os grupos são encorajados a implementar outras opções no menu, como carregamento e armazenamento do estado do jogo da/para memória secundária. (0,1 ponto)
- Os grupos são encorajados a implementar o controle das maiores pontuações do jogo (*ranking*) – se for cabível (para jogos em que isso não é cabível, pode-se criar um *ranking* baseado no tempo decorrido até vencer o jogo). Deve-se registrar o nome do jogador vencedor ao final de cada partida para concorrer ao pódio. (0,1 ponto)
- Os grupos são encorajados a implementar um algoritmo “esperto” para jogadas do computador e/ou automática. (0,1 ponto)
- Os grupos podem propor a utilização de outras estruturas de dados para resolver os problemas de cada jogo. *Contatem o professor antes de implementar essas estruturas*. (0,5 ponto)

Figura 1: Interface simples para o jogo Uno implementada com `raylib`.

Entrega

O trabalho será entregue em **3 (três) etapas**:

1. Formação dos grupos: até o dia **16 de Janeiro de 2024** às 23:59 horas pelo ambiente virtual Moodle.
   - Cada grupo deve **sinalizar** seus integrantes no Fórum "Trabalho Final - Definição dos Grupos" no Moodle. Os grupos devem ser formados por dois ou três integrantes. Se, eventualmente, um aluno não tiver um grupo, este pode ingressar em um grupo existente desde que em comum acordo com os demais membros.
2. Entrega do código: até o dia **14 de Fevereiro de 2024** às 23:59 horas pelo ambiente virtual Moodle.
   - Cada grupo deve fazer o *upload* de um **único arquivo compactado (.zip)** na Atividade "Trabalho Final - Entrega dos Códigos" no Moodle. O arquivo deve ter nome `TF-<nome_de_cada_integrante>-versao_<X>.zip`. Este arquivo deve conter o **código completo e documentado** (arquivos .c e .h de todos os TADs desenvolvidos). O envio de documentos auxiliares que expliquem textualmente o trabalho ou o processo de desenvolvimento é encorajado.
   - O grupo deve incluir o nome dos autores no cabeçalho do arquivo que contém o programa principal, em forma de comentário.
   - O grupo pode fazer upload de diferentes versões da aplicação. Nomeiem adequadamente as versões, tal que a mais recente seja facilmente identificada.
3. Apresentação: no dia **15 de Fevereiro de 2024** das 15:30 às 17:10 horas.
   - Cada grupo deve apresentar (a) o problema (brevemente), (b) a modelagem da solução proposta e (c) as principais funcionalidades da aplicação desenvolvida. Deve-se explicar como as estruturas de dados resolvem os problemas da aplicação. As funcionalidades devem, idealmente, também ser exibidas durante a execução da aplicação. Todos os integrantes do grupo devem saber explicar os pontos (a), (b) e (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. Vocês podem usar um conjunto de *slides* caso achem adequado.
   - A apresentação do trabalho conta nota na avaliação final.

Avaliação

O Trabalho Final possui um peso de **20%** no cálculo da média final do aluno, representando parte da nota concedida no quesito "Trabalho Final" (o restante se refere às notas das Atividades Teóricas (ATs) e Práticas (APs) e Provas (P01 e P02) desenvolvidas ao longo do semestre). O desenvolvimento e entrega do Trabalho Final são requisitos para obtenção de conceito "A" na disciplina. **A aplicação desenvolvida deve atender a todos os requisitos listados neste enunciado, não deve apresentar erros de compilação e deve executar normalmente.** A aplicação desenvolvida deverá demonstrar os seguintes aspectos que serão avaliados:

1. Habilidade em estruturar aplicações pela decomposição da realidade modelada em TADs. Cada TAD deve ser responsável pela manipulação das variáveis de controle das estruturas e propor uma série de funcionalidades (operações/funções). (2,5 pontos)
2. Documentação, legibilidade e modularização do código. (1,5 ponto)
3. Corretude na utilização dos conceitos estudados e uso coerente (otimizado) da(s) estrutura(s) de dados selecionada(s). (2,5 pontos)
4. Formatação e controle de entrada e saída com a construção de interfaces que orientem corretamente o usuário. Garantia da consistência e coerência dos dados e das operações. (1,5 ponto)
5. Clareza, organização e demonstração de domínio da solução implementada na apresentação final do projeto (apresentação). (2 pontos)

Observações

Este trabalho deve refletir a solução individual do grupo para o problema proposto. Casos de plágio (mais de 75% de similaridade – incluindo entregas de outros semestres) serão tratados com severidade e resultarão em anulação da nota do Trabalho Final (vide programa da disciplina). Para detectar e quantificar o plágio no código, será utilizado o *software* MOSS (`http://theory.stanford.edu/~aiken/moss/`).