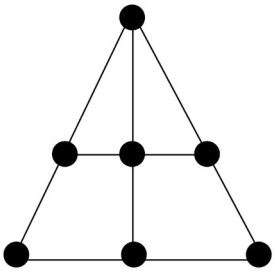
Jogo TRI-angle

O Trabalho Final da disciplina de Laboratório de Programação I consiste na implementação do jogo *TRI-angle*, uma vairação dos conhecidos Jogo da Velha e Jogo Moinho (ou Trilha). O objetivo do jogo é alinhar três peças em um tabuleiro.

Tabuleiro

Trata-se de um jogo de tabuleiro, jogado por 2 pessoas, cada uma com 3 peças de cores diferentes. O jogo é jogado em um tabuleiro em forma de triângulo com pontos de intersecção (casas), onde as peças deverão ser colocadas. A seguir, uma imagem ilustrativa do tabuleiro:



Etapas do Jogo

O jogo possui duas etapas, uma de posicionamento das peças no tabuleiro e uma de movimentação. O jogo inicia com todas as peças fora do tabuleiro. Os jogadores decidem, por sorteio, quem iniciará a partida.

Na primeira etapa, cada jogador, alternadamente, coloca uma peça em uma casa vazia do tabuleiro, até acabarem as peças. Nesta etapa não são permitidos movimentos no tabuleiro.

Os movimentos de jogadas só acontecem quando todas as peças estiverem no tabuleiro. Nesse caso, haverá uma casa vazia. Neste momento, se houver três peças alinhadas, o jogador que as colocou vence a partida. Caso contrário, seguindo a ordem estabelecida por sorteio, cada jogador, alternadamente, faz uma jogada colocando uma de suas peças em uma casa vazia do tabuleiro.

O jogo permite dois tipos de movimentação de peças. No primeiro tipo, uma peça pode ser movida para uma casa adjacente em alinhamento. O segundo tipo de movimento permite que uma peça pule sobre outra peça (do adversário ou não) adjacente a ela, em alinhamento, e seja colocada em uma casa vazia adjacente à peça que foi saltada.

Fim de Jogo

O jogador que conseguir alinhar três peças é o vencedor. Vale lembrar que o jogo não permite captura de peças. O jogo pode durar muito tempo sem que haja um vencedor. Neste caso, os jogadores devem decidir pelo empate.

Implementação

O jogo deve ser implementado em linguagem C, utilizando funções e passagem de parâmetros. A parte gráfica do jogo deve ser implementada com o auxílio da biblioteca Allegro 5. A parte referente à biblioteca gráfica consiste na pesquisa e ambientação com as funções do Allegro 5. Como mencionado em aula, o PET-CC está oferecendo uma oficina sobre a biblioteca Allegro 5 no período de 19 a 22/05. Enquanto isso, o aluno pode instalar a biblioteca e começar a se ambientar. Além disso, toda a lógica de estruturação da lógica do jogo já pode ser planejada e sua implementação iniciada.

O tabuleiro deve ser mapeado em um array bidimensional (implementações com vetor também são possíveis), de modo a retratar o posicionamento das peças. Todas as movimentações devem ser realizadas com o auxílio do mouse, o qual deve indicar a posição inicial e final da peça envolvida no posicionamento inicial das peças (primeira etapa) ou no movimento das peças (etapa 2).

O programa deve oferecer um menu (descrito a seguir), oferecendo ao usuário 2 opções de jogo:

Opção player x player
Essa opção conduz o usuário a uma versão player x player do Jogo. Nesta opção, o programa deve controlar o jogo entre os 2 jogadores, validando jogadas e identificando o momento em que um dos jogadores vence a partida, ou ocorre o empate.

Opção player x computador
Nessa opção, o usuário irá jogar contra o computador. O programa deve conduzir o jogo entre o jogador e o computador, implementando a lógica de "pensamento" do computador. Assim, o programa deve realizar a escolha do posicionamento das peças (primeira etapa do jogo), assim como das movimentações das peças no tabuleiro (segunda etapa do jogo). Esta escolha **não** deve ser aleatória e sim baseada em critérios definidos pelo aluno. Também não devem ser utilizadas técnicas refinadas de inteligência artificial, tais como MCTS (Monte Carlo Tree Search). A ideia é que o aluno crie seus critérios de escolha de jogada para a implementação do "pensamento" do computador.

As orientações a seguir se aplicam a ambas opções de jogo.

Todas as jogadas de cada jogador devem ser validadas pelo programa, de forma a não permitir movimentos inválidos. Em caso de jogada inválida, o programa deve emitir uma mensagem de aviso e solicitar uma nova jogada.

Antes de cada jogada, o programa deve exibir, para cada jogador, todas as possibilidades de movimento, indicando origem e destino de cada jogada válida.

A cada alteração no tabuleiro, o programa deve exibi-lo de forma organizada e de fácil entendimento. Utilize o recurso de limpar a tela a fim de manter o ambiente de jogo organizado.

O programa deve controlar o jogo entre os dois jogadores, identificando o momento de Fim de Jogo e em que situação a partida terminou empatada. Cabe ao programa definir a regra de empate.

Ao final de cada partida, o programa deve oferecer a possibilidade de jogar novamente a mesma versão de jogo ou voltar ao menu.

Menu

O programa deve oferecer um menu contendo as 2 opções de jogo (player x player e player x computador). Além disso, o menu deve oferecer as opções AJUDA, HISTÓRICO e SAIR, conforme segue abaixo.

A opção AJUDA deve fornecer um tutorial sobre o funcionamento do jogo.

A opção HISTÓRICO deve exibir, como o nome sugere, um histórico sobre as partidas anteriores daquele jogador, identificando, para cada opção de jogo (player x player e player x computador) o número de partidas jogadas e em que situação cada uma terminou. Além disso, o histórico deve exibir o maior e o menor tempo decorrido entre as partidas de cada opção de jogo. Para essa informação, o programa deve contar com um *timer* para medir a duração de cada partida. O histórico deve ser salvo em um arquivo, a fim de ser exibido a qualquer momento.

A opção SAIR deve encerrar o jogo.

Recursos

O programa deve fornecer 2 recursos ao jogador: PAUSAR e SALVAR. Estes recursos devem ser exibidos na tela, a fim de que o jogador possa fazer uso.

O recurso PAUSAR, como o nome sugere, implica em parar o jogo e depois reiniciar de onde parou. É importante observar que, neste caso, o *timer* que contabiliza a duração da partida, deve ser pausado e depois reiniciado de onde parou.

O recurso SALVAR deve permitir armazenar o jogo em andamento em um arquivo (diferente do arquivo do Histórico), a fim de retomar em um momento posterior. Somente é possível salvar a última partida por opção de jogo (no caso, a última partida para a qual foi solicitado o salvamento). Sendo assim, o usuário pode ter duas partidas salvas: uma para a opção player x player e outra para a opção player x computador. Para o salvamento de uma partida em andamento, deve-se armazenar, pelo menos: a opção de jogo (player x player e player x computador), o estado do tabuleiro (contendo as posições de todas as peças), o número e cores de peças fora do tabuleiro (para o caso de pausa na primeira etapa de jogo), indicação do próximo jogador e o valor do *timer*. Estas informações devem estar disponíveis para o programa quando o jogador quiser retomar o jogo interrompido. A retomada do jogo pode consistir em mais uma opção do menu descrito anteriormente.

Output

A implementação deve se preocupar com a exibição do andamento do jogo (*output*), de modo a permitir seu acompanhamento de forma organizada e adequada. Neste sentido, o programa deve exibir todo o fluxo de andamento da partida, indicando cada movimentação.

Por exemplo, supondo uma implementação com matriz, em que as linhas são identificadas com letras e as colunas com números. Uma possível exibição de jogada seria o jogador com peças amarelas movimentou a peça B1 para a posição C1, ou jogador com peças vermelhas movimentou a peça C1 para a posição C2. A avaliação irá levar em conta o modo como a exibição do jogo foi implementada.

Segue abaixo o código para limpar a tela:

```
#include <stdlib.h>

#ifdef _WIN32
#define CLEAR "cls"
#else //In any other OS
#define CLEAR "clear"
#endif

void limpaTela() {
  system(CLEAR);
}
```

Orientações:

Além da especificação anterior, que descreve o programa a ser desenvolvido, deve-se considerar que:

- A realização do trabalho é individual.
- Fazem parte do contexto deste trabalho o entendimento do jogo e a interpretação da especificação.
- Na avaliação, as implementações das opções player x player e player x computador possuem pesos diferentes.

- Além da Oficina da Biblioteca Allegro, a disciplina conta com o suporte de um aluno do PET-SI (Gustavo Pott – gpoliveira@inf.ufsm.br) para orientar sobre a instalação e uso da biblioteca gráfica.
- Implementações de recursos adicionais (ex. sons, cores, bitmaps para modo gráfico, etc), em substituição às funcionalidades solicitadas, **não** agregam valor ao jogo.

Implementação:

- Certifique-se de modularizar as funcionalidades do jogo em funções com escopo definido e comportamento adequado (estrutura e organização do código fazem parte da avaliação).
  - O código deve priorizar a otimização, de forma a evitar o uso de trechos redundantes, os quais podem ser modularizados (funções).
  - Deve ser usada passagem de parâmetros (no lugar de variáveis globais), sempre que possível.
- Decisões de implementação serão avaliadas e distinguem uma implementação das demais.
- Não devem ser inseridos comentários no código.
- O código do jogo deve estar devidamente indentado.
- Observar para que seja utilizado o mínimo possível de funções/comandos específicos do Windows, uma vez que o trabalho será corrigido na plataforma Linux. Quando necessário, deve ser exibido, sob a forma de comentário, o que deve ser alterado e como deve ser alterado. Os comentários são permitidos apenas nessa situação.
- Os monitores da disciplina (Ivan Martignago – immartignago@inf.ufsm.br e Giordana Camargo – gccamargo@inf.ufsm.br) estão orientados a apenas oferecer suporte para que os alunos desenvolvam o trabalho. O desenvolvimento em si deve ser realizado exclusivamente pelo aluno.

Apresentação:

- Todos os trabalhos devem ser apresentados.
- As apresentações envolvem a explicação das funcionalidades implementadas.
- As apresentações serão realizadas no notebook do aluno. Caso não possua, deve providenciar um emprestado ou utilizar os computadores do Laboratório. É de responsabilidade do aluno a instalação do jogo (e do Allegro 5, se for o caso) previamente à data da apresentação. Para o uso do laboratório, verifique com antecedência se a biblioteca Allegro 5 está instalada.
  - O aluno deve se certificar de que tudo esteja funcionando para a apresentação, evitando situações do tipo “No meu computador funciona” ou “Até hoje de manhã funcionava”...
- O cronograma das apresentações será publicado no Moodle no dia **09/06/2025**. As apresentações iniciam no dia **10/06/2025**.

Avaliação:

- Serão avaliadas todas as peculiaridades solicitadas na especificação, além da criatividade do aluno na implementação da solução.
- Durante a apresentação, o aluno será questionado sobre a implementação do trabalho e deve estar ciente de todos os seus detalhes.
- Trabalhos não apresentados serão desconsiderados e o aluno obterá nota zero.
- Serão utilizadas ferramentas de análise de similaridade de código-fonte (p. ex. MOSS, JPlag). Em caso de cópia de trabalhos, o aluno será penalizado.
- Aconselha-se que os alunos não troquem informações entre si, de modo a evitar coincidências entre os códigos, as quais serão acusadas pelas ferramentas de análise. Quaisquer similaridades terão impacto na avaliação do trabalho.
- A utilização de código de outros trabalhos que não sejam de autoria do aluno é considerada cópia/plágio e está sujeita às determinações da Resolução 017/2018.
- Não serão analisados códigos .c antes do período de entrega.

Entrega:

- Data: até às 23h59m de **08/06/2025** (impreterivelmente!).
- Aos alunos que costumam fazer alterações no código pouco antes do horário de entrega, aconselha-se a fazer o envio de uma versão preliminar, a fim de não perder o prazo por algum motivo.
  - Não serão aceitos trabalhos fora do prazo especificado, nem mesmo com desconto de nota.
- Formato: nomeAluno.zip, o qual **não** deve conter arquivos executáveis.
- Envio: deve ser postado no Moodle, obedecendo data e horário limites. Não será aceita qualquer outra forma de envio.