Definição do trabalho da M2

Data de entrega: 16/05/2024 – Enviar código-fonte no AVA Univali (até 18h59min) e apresentar em aula para o professor (10 minutos).

Modalidade: em Trio ou Dupla.

Visão Geral:

O Mastermind (no Brasil, Senha) é um jogo de tabuleiro inventado por Mordechai Meirowitz e distribuído inicialmente pela Invicta Plastics. Publicado em 1971, o jogo vendeu mais de 50 milhões de tabuleiros em 80 países, tornando-se o mais bem sucedido novo jogo da década de 1970. Atualmente, no Brasil é vendido pela Grow com o tabuleiro preto e cinza, e os pinos do jogo em azul, amarelo, verde, vermelho, rosa, roxo e laranja (Wikipedia, 2021).

Descrição:

Um jogo de Mastermind tem pinos de sete cores diferentes, aleatórias, exceto preto e branco. Os pinos pretos e brancos são menores. Há quatro buracos grandes em cada fileira, em 10 fileiras, uma abaixo da outra. E ao lado delas, um quadrado menor, com quatro buracos menores, dois em cima de dois. Uma fileira, que seria a décima primeira, tem um defletor que esconde seus buracos. O desafiador faz uma combinação com quatro pinos coloridos, sem repetir as cores de cada pino, e as põe na décima primeira fileira e levanta o defletor, escondendo a senha. Então, o desafiado tenta adivinhar a senha, pondo quatro pinos que ele acha que são a senha na primeira fileira, e o desafiador põe os pinos pretos e brancos no quadrado menor ao lado. A regra dos pinos pretos e brancos são essas: o branco significa que há uma cor certa no lugar errado, e o preto significa que há uma cor certa no lugar certo, e nenhum pino significa que uma das cores não é contida na senha. O desafiado vai tentando adivinhar, se guiando pelos pinos pretos e brancos. Se o desafiado não acertar até a 10ª fileira, o desafiador fecha o defletor e revela a senha, mas se adivinhar, o desafiador põe quatro pinos pretos e revela a senha.

REGRAS PARA O DESENVOLVIMENTO

O jogo deverá possuir um menu onde será possível escolher:

- Jogar
- Dificuldade
- Sobre
- Fim

A seguir serão listadas, de trás para frente, o que deve ser implementado em cada parte do menu.

Fim: Essa é uma opção para finalizar o programa. Observe que seu jogo só deve ser encerrado ao selecionar essa opção, caso qualquer outra opção seja escolhida ela deve executar e voltar ao menu.

Sobre: Quando essa opção for selecionada, informe: a equipe de desenvolvimento (o nome de cada membro da equipe), o mês/ano (exemplo: maio/2024) e o nome do professor/disciplina.

Dificuldade: Quando essa opção for selecionada, o jogador poderá escolher entre 3 dificuldades:

- Fácil: nessa dificuldade o jogador terá que acertar uma sequência de 3 dígitos e terá 08 tentativas para isso.
- Média: nessa dificuldade o jogador terá que acertar uma sequência de 4 dígitos e terá 10 tentativas para isso. Essa é a dificuldade padrão, ou seja, se não for alterada, é nessa dificuldade que o jogo irá rodar.
- Difícil: nessa dificuldade o jogador terá que acertar uma sequência de 5 dígitos e terá 12 tentativas para isso.

Jogar: Essa é a parte onde o jogo acontece de verdade!

Inicie o jogo gerando os números aleatórios que serão o código que o jogador tentará quebrar. Esse código deverá respeitar as seguintes regras:

- Não deverão ser gerados números repetidos; e
- Os dígitos gerados deverão estar entre 1 e 6 (`1 <= x <= 6`).

Uma vez escolhido o código a ser descoberto pelo jogador, você deverá solicitar ao jogador uma entrada de acordo com a dificuldade do jogo. (ex. 4 valores para dificuldade média)

Após cada tentativa o jogador deverá ser informado:

- O número de tentativas restantes (muda de acordo com a dificuldade)
- Quantos foram os números digitados que estão corretos e estão na posição correta. (note que você diz quantos e não quais)
- Quantos foram os números digitados que estão corretos, mas não estão nas posições corretas.

O jogo deve acabar quando o jogador acertar todos os números na ordem correta ou quando o número de tentativas zerar.

Se o jogador vencer, informe que ele venceu e retorne ao menu.

Se o jogador perder, informe que ele perdeu e retorne ao menu.

Obs.: Para o desenvolvimento do código não poderão ser utilizadas variáveis compostas (*arrays*) e nem funções.

Dicas de desenvolvimento:

O código, a seguir, exemplifica o uso das funções `rand()` e `srand()`;

- `rand()` gera um número pseudo-aleatório entre `0` e `RAND_MAX`, mas essa faixa pode ser facilmente alterada com o operador de resto da divisão inteira.
- `srand()` gera uma nova semente aleatória baseada no parâmetro passado entre os parênteses da função. É comum utilizar a função `time()`, pois ela pega o horário do sistema que muda a cada milésimo de segundo. Note que se a função `srand()` não for utilizada a sequência de números pseudo-aleatórios gerados pela função `rand()` será sempre a mesma.

```
#include <iostream>
#include <ctime> /// Usada para habilitar a função time()
using namespace std;

int main() {
    int r, i;

    /// Os números gerados serão sempre iguais, pois usam a mesma semente de geração
    cout << "Gerando 10 valores pseudo-aleatórios entre 0 e 99 (sempre iguais):\n";
    for(i = 0; i < 10; i++) {
        r = rand() % 100; // O resto serve para limitar a faixa de valores
        cout << r << '\t';
    }

    /// Os números gerados serão diferentes a cada execução, pois a semente será diferente.
    cout << "\nGerando 10 valores pseudo-aleatórios entre 0 e 99 (diferentes):\n";
    srand(time(NULL)); // Comando para gerar a semente com base no horário do sistema
    for(i = 0; i < 10; i++) {
        r = rand() % 100; // O módulo coloca os números gerados entre 0 e divisor - 1
        cout << r << '\t';
    }
}
```

Outros dois comandos bastante úteis no desenvolvimento de programas no console, são os comandos `system("cls")` e `system("pause")`.

- `system("cls")` é um comando que limpa a tela do console (**cl**ear **s**creen). Esse comando é bastante útil, pois em uma tela limpa é mais fácil dar destaque aquilo que se está mostrando no momento.
- `system("pause")` é um comando útil, principalmente quando usado em conjunto com o `system("cls")`, pois ele pausa a execução da aplicação até que o usuário aperte qualquer tecla, bastante útil quando se quer exibir algo antes de limpar a tela para iniciar uma nova execução.

***Apresentação em aula* (TRIO ou DUPLA) – 10 minutos por grupo. Critérios:**

- *Domínio de cada integrante sobre o trabalho;*
- *Cumprimento do tempo;*
- *Arguição nas perguntas;*
- *Sequência e Organização da apresentação.*

***Entrega**:*

- ***Postagem do código-fonte no AVA Univali - Trabalho T2***

Critérios de Avaliação:

---

1. Organização e clareza do código = 20% da nota.
2. Comentários pertinentes e oportunos no código = 10% da nota.
3. Funcionamento correto conforme a especificação = 40% da nota.
4. Apresentação = 30% da nota.

Obs.: Todas as notas relativas ao código dependem do desempenho na apresentação. Sem apresentação o trabalho terá nota ZERO.

Trabalhos que apresentarem **similaridade de código-fonte** terão a nota dividida pelo número de entregas similares. Por exemplo, se 3 grupos enviarem código-fonte parecidos, a nota desses grupos será dividida por 3.