

*Definição do trabalho da M3**JOGO DA MEMÓRIA*

Data de entrega: 30/06/2022. (até 19:00)

Modalidade: grupos de 3 ou 4 integrantes.

Visão Geral:

1	4	5	2
7	2	8	7
3	6	1	4
6	5	3	8

O jogo da memória é um clássico jogo formado por peças que apresentam uma figura em um dos lados. Cada figura se repete em duas peças diferentes. Para começar o jogo, as peças são postas com as figuras voltadas para baixo, para que não possam ser vistas. O jogador deve, na sua vez, virar duas peças. Caso as figuras sejam iguais, ele recolhe consigo esse par. Se forem peças diferentes, estas são viradas novamente, e a vez é passada ao participante seguinte. Ganha o jogo quem tiver descoberto mais pares, quando todos eles tiverem sido recolhidos.

Bom jogo!

REGRAS PARA O DESENVOLVIMENTO

O jogo deverá, inicialmente, definir uma matriz principal com todas as posições preenchidas, isso porque o jogo irá utilizar essa matriz principal para gerar a matriz jogo.

A matriz deve ser composta por no mínimo 16 peças (caractere ou número) (8 pares, 4x4). Inicie o jogo, aplicando, **aleatoriamente**, uma das seguintes operações na matriz principal para gerar a matriz gabarito (a matriz gabarito deverá ser preenchida durante a execução do algoritmo, ou seja, não é permitido ter ela já disponível no algoritmo):

1. **Sem modificação:** a matriz gabarito será uma cópia da matriz principal.
2. **Transposta:** obtida transportando-se ordenadamente os elementos das linhas da matriz principal para as colunas da matriz gabarito.

MATRIZ PRINCIPAL	MATRIZ TRANSPOSTA																																
<table><tr><td>1</td><td>4</td><td>5</td><td>2</td></tr><tr><td>7</td><td>2</td><td>8</td><td>7</td></tr><tr><td>3</td><td>6</td><td>1</td><td>4</td></tr><tr><td>6</td><td>5</td><td>3</td><td>8</td></tr></table>	1	4	5	2	7	2	8	7	3	6	1	4	6	5	3	8	<table><tr><td>1</td><td>7</td><td>3</td><td>6</td></tr><tr><td>4</td><td>2</td><td>6</td><td>5</td></tr><tr><td>5</td><td>8</td><td>1</td><td>3</td></tr><tr><td>2</td><td>7</td><td>4</td><td>8</td></tr></table>	1	7	3	6	4	2	6	5	5	8	1	3	2	7	4	8
1	4	5	2																														
7	2	8	7																														
3	6	1	4																														
6	5	3	8																														
1	7	3	6																														
4	2	6	5																														
5	8	1	3																														
2	7	4	8																														

--	--

3. **Invertida por linha:** obtida transportando-se ordenadamente os elementos das últimas linhas da matriz principal para as primeiras linhas da matriz gabarito.

MATRIZ PRINCIPAL	MATRIZ INVERTIDA POR LINHA																																
<table><tr><td>1</td><td>4</td><td>5</td><td>2</td></tr><tr><td>7</td><td>2</td><td>8</td><td>7</td></tr><tr><td>3</td><td>6</td><td>1</td><td>4</td></tr><tr><td>6</td><td>5</td><td>3</td><td>8</td></tr></table>	1	4	5	2	7	2	8	7	3	6	1	4	6	5	3	8	<table><tr><td>6</td><td>5</td><td>3</td><td>8</td></tr><tr><td>3</td><td>6</td><td>1</td><td>4</td></tr><tr><td>7</td><td>2</td><td>8</td><td>7</td></tr><tr><td>1</td><td>4</td><td>5</td><td>2</td></tr></table>	6	5	3	8	3	6	1	4	7	2	8	7	1	4	5	2
1	4	5	2																														
7	2	8	7																														
3	6	1	4																														
6	5	3	8																														
6	5	3	8																														
3	6	1	4																														
7	2	8	7																														
1	4	5	2																														

4. **Invertida por coluna:** obtida transportando-se ordenadamente os elementos das últimas colunas da matriz principal para as primeiras colunas da matriz gabarito.

MATRIZ PRINCIPAL	MATRIZ INVERTIDA POR COLUMNA																																
<table><tr><td>1</td><td>4</td><td>5</td><td>2</td></tr><tr><td>7</td><td>2</td><td>8</td><td>7</td></tr><tr><td>3</td><td>6</td><td>1</td><td>4</td></tr><tr><td>6</td><td>5</td><td>3</td><td>8</td></tr></table>	1	4	5	2	7	2	8	7	3	6	1	4	6	5	3	8	<table><tr><td>2</td><td>5</td><td>4</td><td>1</td></tr><tr><td>7</td><td>8</td><td>2</td><td>7</td></tr><tr><td>4</td><td>1</td><td>6</td><td>3</td></tr><tr><td>8</td><td>3</td><td>5</td><td>6</td></tr></table>	2	5	4	1	7	8	2	7	4	1	6	3	8	3	5	6
1	4	5	2																														
7	2	8	7																														
3	6	1	4																														
6	5	3	8																														
2	5	4	1																														
7	8	2	7																														
4	1	6	3																														
8	3	5	6																														

Após a geração da matriz gabarito deverá ser criada a matriz jogo (com todas as posições vazias). Essa matriz é a matriz que será exibida e utilizada para a marcação dos pares descobertos.

Uma vez que a matriz gabarito e a matriz jogo estão preparadas o jogador poderá iniciar as jogadas respeitando as seguintes regras:

- Haverá apenas um jogador jogando sozinho, "contra o programa" .
- Em cada jogada, o jogador informa a posição (linha e coluna) das peças a serem "viradas"(mostradas).
- O programa deve mostrar o valor destas peças. Caso as peças tenham o mesmo valor, estas ficam indisponíveis e aparece a mensagem "JOGADA OK". Caso contrário, aparece a mensagem "JOGADA NOK".
- O jogador pode fazer um número de jogadas igual ao triplo dos pares. Por exemplo, para 8 pares, 24 jogadas
- O jogo encerra quando o jogador encerrar o número de jogadas ou até ele descobrir todos os pares.
- Durante o jogo, o programa deve mostrar quantidade de jogadas realizadas pelo jogador.
- No encerramento deve ser apresentada uma mensagem de vitória ou derrota.

Obs.: Para o desenvolvimento do código não poderão ser utilizadas funções ou structs.

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.

```
1  #include <iostream>
2  #include <time.h> //para habilitar a função time
3  using namespace std;
4
5  int main()
6  {
7      srand(time(NULL)); //semente randomica gerada a partir da hora do sistema
8
9      int numeroAleatorio;
10
11     numeroAleatorio = rand()%10; //0 %(mod) coloca os números gerados entre 0 e o resto da divisão-1
12
13     cout << numeroAleatorio << endl;
14
15 }
```

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 (**clear screen**). 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.

*Os comandos equivalentes ao `system("cls")` e `system("pause")` no linux/MacOS são respectivamente o `system("clear")` e `system("read 0 -p")`.

Defesa (Obrigatória)

Durante a defesa serão realizados questionamento sobre o trabalho realizado pelo grupo. A defesa é obrigatória e deverá ser feita pelos integrantes do grupo na aula. Se algum integrante não estiver presente durante a aula de defesa, deverá justificar a falta, o mesmo defenderá posteriormente em data a ser agendada com o professor.

Entregas:

- Postar no repositório criado especialmente para o trabalho no material didático: **Trabalho T3**
- Código fonte desenvolvido: é de responsabilidade do grupo verificar se o arquivo postado é o correto.

Critérios de Avaliação:

1. Organização e clareza do código = 5% da nota.
2. Identificação dos autores e Comentários pertinentes e oportunos no código = 10% da nota.
3. Funcionamento correto conforme a especificação = 40% da nota.
4. Recursos da linguagem utilizados = 20% da nota.
5. Apresentação/Defesa do código = 25% da nota.

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