
Definição do trabalho da M2

Data de entrega: 29/10/2021. (até 08h)

Modalidade: grupos de 3 ou 4 integrantes.

Visão Geral:



O jogo da forca é um jogo em que o jogador tem que acertar qual é a palavra proposta, tendo como dica o número de letras e o tema ligado à palavra. A cada letra errada, é desenhado uma parte do corpo do enforcado. O jogo termina ou com o acerto da palavra ou com o término do preenchimento das partes corpóreas do enforcado (Wikipedia, 2021).

Descrição:

Para começar o jogo se desenha uma base e um risco correspondente ao lugar de cada letra.

Por exemplo, para a palavra "MERCADO", se escreve: M E R C A D O -----> _ _ _ _ _

O jogador que tenta adivinhar a palavra deve ir dizendo as letras que podem existir na palavra. Cada letra que ele acerta é escrita no espaço correspondente: M E R C A D O → M _ _ C A _ _

Caso a letra não exista nessa palavra, ele perde uma tentativa (desenha-se uma parte do corpo, iniciando pela cabeça, tronco, braços...)

O jogo é ganho se a palavra é adivinhada. Caso o jogador não descubra qual palavra até esgotar o número de tentativas ele perde.

REGRAS PARA O DESENVOLVIMENTO

O jogo deverá possuir um menu onde será possível escolher:

- Jogar
- Sobre
- Fim

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

Fim:

Essa é a 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 retornar ao menu no fim de sua execução.

Sobre:

Quando essa opção for selecionada, deverão ser exibidas a equipe de desenvolvimento (o nome de cada membro da equipe), o mês/ano (exemplo: outubro/2021) e o nome do professor/disciplina. Também deve ser exibida a regra do uso das letras, por exemplo: usar somente letras maiúsculas nas tentativas.

Jogar:

Essa é a parte onde o jogo acontece de verdade!

Inicie o jogo escolhendo aleatoriamente entre 10 palavras que serão o código que o jogador tentará quebrar. Esse código deverá respeitar as seguintes regras:

- A cada nova partida uma palavra diferente deve ser selecionada; e
- As palavras devem conter 6 caracteres representados por 6 variáveis do tipo char
- Uma vez escolhida a palavra que deverá ser descoberta pelo jogador, você deverá solicitar que o jogador digite uma letra
- O jogador tem 10 tentativas para acertar a palavra completa (todos os seis caracteres)
- Após cada tentativa o jogador deverá ser informado:
 - O número de tentativas restantes
 - O número acertos
 - Os acertos realizados nas posições corretas (Ex: Se a palavra sorteada for PENSAR, ao digitar a letra A, deve ser exibido: _ _ _ _ A _ .

O jogo deve acabar quando o jogador acertar todas as letras 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 funções, com exceção das apresentadas nas Dicas de desenvolvimento.

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      int r, i;
8      ///os números gerados serão sempre iguais, pois usam sempre a mesma semente randômica
9      cout << "\nGerando 10 valores randômicos entre 0 e 99 (sempre iguais): \n";
10     for(i=0;i<10;i++){
11         r=rand()%100; ///o % coloca os números gerados entre 0 e o resto da divisão-1
12         cout<<r<<" ";
13     }
14     ///os números gerados mudarão a cada execução, pois estamos criando uma nova semente
15     cout << "\nGerando 10 valores randômicos entre 0 e 99: \n";
16     srand(time(NULL)); ///semente randômica gerada a partir da hora do sistema
17     for(i=0;i<10;i++){
18         r=rand()%100; ///o % coloca os números gerados entre 0 e o resto da divisão-1
19         cout<<r<<" ";
20     }
21     return 0;
22 }

```

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.

Defesa (Obrigatória):

Durante a defesa serão realizados questionamentos sobre o trabalho realizado pelo grupo a um dos integrantes. O integrante que realizará a defesa será sorteado para representar o grupo e o seu desempenho refletirá na nota de todos os integrantes do grupo. A defesa é obrigatória e deverá ser feita em aula na data de entrega. Se algum integrante não estiver presente durante a aula de defesa e for o sorteado, outro integrante será sorteado para representar o grupo. Porém o integrante faltante terá que realizar a defesa posteriormente para validar sua nota e a nota do grupo será ajustada conforme seu desempenho.

Entregas:

- *Postar no repositório criado especialmente para o trabalho no BlackBoard: **Postagem Trabalho T2***
- *Código desenvolvido em C++ no CodeBlocks ou outra IDE (anexar arquivo de texto simples ".txt" e postar).*

Cr terios de Avalia  o:

- 1. Organiza  o e clareza do c digo = 5% da nota.*
- 2. Identifica  o dos autores e Coment rios pertinentes e oportunos no c digo = 10% da nota.*
- 3. Funcionamento correto conforme a especifica  o = 40% da nota.*
- 4. Recursos da linguagem utilizados = 20% da nota.*
- 5. Apresenta  o do c digo = 25% da nota.*

Obs.: Todas as notas relativas ao c digo dependem do desempenho na defesa.