**PROJETO DE TÉCNICAS E DESENVOLVIMENTO DE ALGORITMOS PARA A 2ª UNIDADE;  
  
Participantes do Grupo:**

Brenno Lucas Sabino da Silva | RGM: 37663682

Eduardo Costa Dias Lins de Albuquerque | RGM: 38223538

Guilherme Mendonça Silva | RGM: 38430061

Zeus Fontes Barbosa | RGM: 37300873

* Introdução

O jogo da forca, funciona da seguinte maneira: um dos participantes escolhe uma palavra e desenha espaços em branco representando cada letra dessa palavra. O objetivo dos outros jogadores é adivinhar as letras que compõem a palavra. Tentativas incorretas resultam em uma parte do desenho de uma figura de um boneco sendo completada. O jogo termina quando os jogadores acertam a palavra e vencem ou quando o desenho do boneco é completado, indicando a derrota.

* Resultados

Sobre a função das bibliotecas:

#include <stdlib.h> : Fornece funções gerais, como alocação de memória, conversão de tipos, geração de números aleatórios, e controle de processos.

#include <string.h> : Contém funções para manipulação de strings, como cálculo de tamanho, cópia, concatenação, comparação, e busca de caracteres ou substrings.

#include <locale.h> : usada para ajustar o idioma.

#include <time.h> : Fornece funções relacionadas ao tempo e datas, como obter a hora atual, manipular e formatar tempo, e calcular diferenças de tempo.

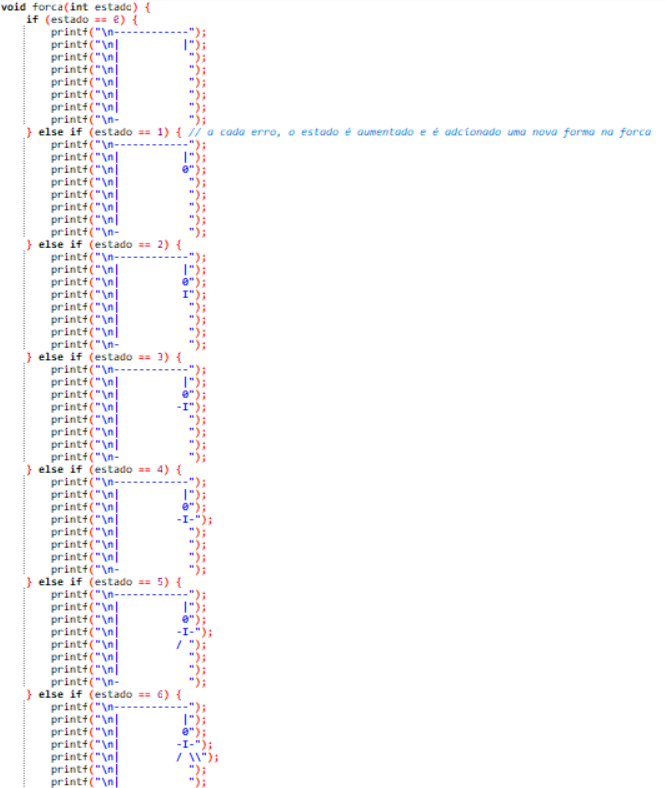
Para servir como ponto de partida para o nosso jogo obviamente seria necessário criar uma interface para que o usuário possa interagir.

Criamos **5** opções para o usuário escolher, uma para jogar, outra para ver o Ranking dos jogadores, outra para cadastrar as palavras que serão utilizadas no jogo, outra para os créditos do jogo e por fim uma opção para fechar o programa.



Após isso criamos uma função para imprimir na tela o nosso jogo chamada de “forca()”, caracterizado por uma forca e um espaço onde o boneco vai se formando a cada erro. Ao todo serão 7 estágios até o boneco ser totalmente imprimido na tela, indicando fim de jogo.

No último foi necessário inserir ”\“ duas vezes pois se trata de um caractere de escape, portanto a única forma de exibi-lo na tela é inserindo-o dessa maneira ”\\”.



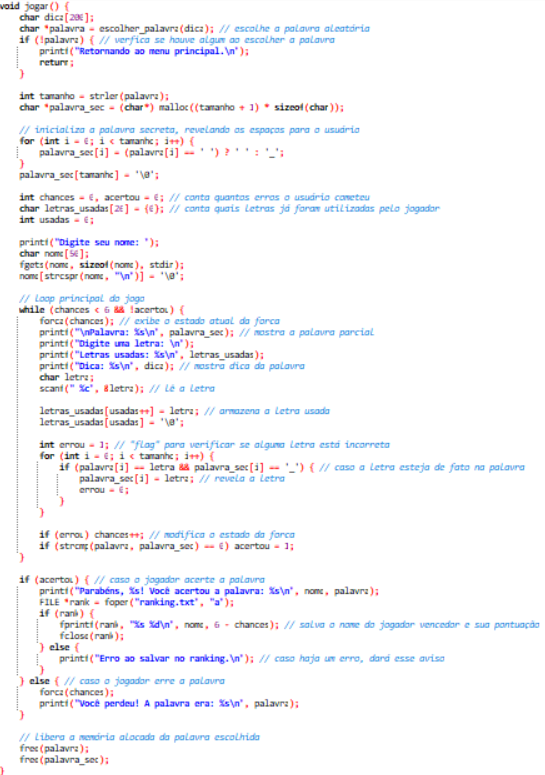
Após isso criamos mais uma função chamada “cadastrar\_palavras()” para cadastrar as palavras que serão utilizadas no jogo. Essa função cria ou abre caso já exista um arquivo.txt para armazenar as palavras de até 100 caracteres e as dicas de até 200 caracteres que estão associadas a essas palavras e criamos um laço de repetição usando do while para o usuário cadastrar palavras até que ele digite “>sair”, saindo do laço de repetição e retornando para a interface de início do programa já com as palavras e dicas registradas.

Também criamos uma função chamada “escolher\_palavra()” que abre o arquivo .txt em modo de leitura e lê até o final do arquivo em busca das palavras registradas (caso não haja nenhuma palavra registrada o jogo não vai iniciar), escolhe uma palavra aleatória separando ela e a dica

A partir daí a lógica do jogo finalmente tem início através da função “jogar()” que chama a função “escolher\_palavra()”, verifica se há algum erro ao escolher a palavra, inicializa a palavra secreta, revelando os espaços para o usuário, conta quantos erros o usuário cometeu através de duas variáveis chamadas “chances” e “acertou”, conta quais letras já foram utilizadas pelo jogador através de uma string de 26 caracteres chamada “letras\_usadas” e cadastra o nome do jogador em uma string de até 50 caracteres.

Após isso o loop principal do jogo inicia. exibindo o estado atual da forca com base na variável “chances” imprime na tela a dica, a palavra parcial e lê a letra que o usuário digitar. Caso alguma letra esteja correta a letra será revelada na palavra. Se a letra for incorreta a variável “chances” recebe + 1, alterando o estado da forca. Caso o jogador acerte a palavra será criado ou aberto o arquivo ranking.txt salvando o nome do jogador vencedor e sua pontuação. Caso o jogador erre a palavra será imprimida na tela a derrota e qual era a palavra.

Após isso é liberada a memória alocada da palavra escolhida.



Fizemos obviamente uma função chamada “exibir\_ranking()” que abre o arquivo .txt de ranking em modo leitura, verifica se o arquivo não existe ou não pôde ser aberto e imprime na tela o nome dos jogadores. Também foi feita outra função chamada “exibir\_creditos()” imprimido na tela o nome dos desenvolvedores.

* Código

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

#include <time.h>

typedef struct {

char nome[50];

int pontuacao;

} Jogador; // armazena o nome e a pontuação do player

// funções

void forca(int estado);

void jogar();

void exibir\_ranking();

void cadastrar\_palavras();

void exibir\_creditos();

char\* escolher\_palavra(char \*dica); // escolhe uma palavra aleatória do arquivo e retorna a dica

// menu inicial; contém todas as opções para o usuário interagir

void menu() {

int opcao;

do {

printf("\n=========== Menu ===========\n");

printf("1. Jogar\n");

printf("2. Ver Ranking\n");

printf("3. Cadastrar Palavras\n");

printf("4. Créditos\n");

printf("5. Sair\n");

printf("============================\n");

printf("Escolha uma opção: ");

scanf("%d", &opcao);

getchar(); // limpa o buffer de entrada

// escolha das opções

switch (opcao) {

case 1:

jogar();

break;

case 2:

exibir\_ranking();

break;

case 3:

cadastrar\_palavras();

break;

case 4:

exibir\_creditos();

break;

case 5:

printf("Saindo do jogo. Até mais!\n");

break;

default:

printf("Opção inválida! Tente novamente.\n"); // prevenção de erros de usuário

}

} while (opcao != 5);

}

// montagem de um desenho de forca para o usuário

void forca(int estado) {

if (estado == 0) {

printf("\n------------");

printf("\n| |");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 1) { // a cada erro, o estado é aumentado e é adcionado uma nova forma na forca

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 2) {

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| I");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 3) {

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| -I");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 4) {

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| -I-");

printf("\n| ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 5) {

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| -I-");

printf("\n| / ");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} else if (estado == 6) {

printf("\n------------");

printf("\n| |");

printf("\n| 0");

printf("\n| -I-");

printf("\n| / \\");

printf("\n| ");

printf("\n| ");

printf("\n- ");

} // o último estagio mostra a forca já completa, indicando o "game over"

}

// sistema de cadastro de palavras

void cadastrar\_palavras() {

printf("Digite uma nova palavra e sua dica (ou '>sair' para finalizar):\n");

//cria (ou abre caso já exista) o .txt para armazenar as palavras

FILE \*arquivo = fopen("palavras registradas.txt", "a");

if (!arquivo) {

printf("Erro ao abrir o arquivo.\n");

return;

}

char palavra[100], dica[200];

// possibilita vários cadastros

do {

printf("\nPalavra: ");

fgets(palavra, sizeof(palavra), stdin);

palavra[strcspn(palavra, "\n")] = '\0'; // remove o caractere "\n" capturado pelo fgets

if (strcmp(palavra, ">sair") == 0) break; // sai do loop do cadastro caso o usuário digite ">sair"

printf("Dica para a palavra '%s': ", palavra);

fgets(dica, sizeof(dica), stdin);

dica[strcspn(dica, "\n")] = '\0'; // remove o "\n" da dica também

fprintf(arquivo, "%s;%s\n", palavra, dica); // salva tanto a palavra quanto a dica no arquivo .txt

} while (1);

fclose(arquivo);

printf("\nPalavras e dicas cadastradas com sucesso.\n"); // confirma que o processo de cadastro foi bem sucedido

}

// função de aleatoriedade na escolha das palavras

char\* escolher\_palavra(char \*dica) {

// abre o .txt em modo leitura

FILE \*arquivo = fopen("palavras registradas.txt", "r");

if (!arquivo) {

printf("Erro: Arquivo de palavras não encontrado.\n");

return NULL;

}

fseek(arquivo, 0, SEEK\_END); // lê até o final do arquivo em busca das palavras

if (ftell(arquivo) == 0) {

printf("Nenhuma palavra registrada. Cadastre palavras antes de jogar.\n");

fclose(arquivo);

return NULL; // impede que o jogo seja iniciado sem nenhuma palavra

}

rewind(arquivo);

char linha[300];

int total\_palavras = 0;

while (fgets(linha, sizeof(linha), arquivo)) total\_palavras++;

srand(time(NULL)); // geração de números aléatorios

int escolha = rand() % total\_palavras; // escolha aleatória da palavra

rewind(arquivo); // retorna o ponteiro para o início do arquivo

int contador = 0;

char \*palavra = (char\*) malloc(100 \* sizeof(char));

// lê o arquivo até encontrar a palavra escolhida

while (fgets(linha, sizeof(linha), arquivo)) {

if (contador == escolha) {

linha[strcspn(linha, "\n")] = '\0'; // remove o "\n"

char \*token = strtok(linha, ";"); // separa palavra e dica

strcpy(palavra, token);

token = strtok(NULL, ";");

if (token) strcpy(dica, token);

break;

}

contador++;

}

fclose(arquivo); // fecha o arquivo

return palavra;

}

// inicio da lógica do jogo da forca

void jogar() {

char dica[200];

char \*palavra = escolher\_palavra(dica); // escolhe a palavra aleatória

if (!palavra) { // verfica se houve algum ao escolher a palavra

printf("Retornando ao menu principal.\n");

return;

}

int tamanho = strlen(palavra);

char \*palavra\_sec = (char\*) malloc((tamanho + 1) \* sizeof(char));

// inicializa a palavra secreta, revelando os espaços para o usuário

for (int i = 0; i < tamanho; i++) {

palavra\_sec[i] = (palavra[i] == ' ') ? ' ' : '\_';

}

palavra\_sec[tamanho] = '\0';

int chances = 0, acertou = 0; // conta quantos erros o usuário cometeu

char letras\_usadas[26] = {0}; // conta quais letras já foram utilizadas pelo jogador

int usadas = 0;

printf("Digite seu nome: ");

char nome[50];

fgets(nome, sizeof(nome), stdin);

nome[strcspn(nome, "\n")] = '\0';

// loop principal do jogo

while (chances < 6 && !acertou) {

forca(chances); // exibe o estado atual da forca

printf("\nPalavra: %s\n", palavra\_sec); // mostra a palavra parcial

printf("Digite uma letra: \n");

printf("Letras usadas: %s\n", letras\_usadas);

printf("Dica: %s\n", dica); // mostra dica da palavra

char letra;

scanf(" %c", &letra); // lê a letra

letras\_usadas[usadas++] = letra; // armazena a letra usada

letras\_usadas[usadas] = '\0';

int errou = 1; // "flag" para verificar se alguma letra está incorreta

for (int i = 0; i < tamanho; i++) {

if (palavra[i] == letra && palavra\_sec[i] == '\_') { // caso a letra esteja de fato na palavra

palavra\_sec[i] = letra; // revela a letra

errou = 0;

}

}

if (errou) chances++; // modifica o estado da forca

if (strcmp(palavra, palavra\_sec) == 0) acertou = 1;

}

if (acertou) { // caso o jogador acerte a palavra

printf("Parabéns, %s! Você acertou a palavra: %s\n", nome, palavra);

FILE \*rank = fopen("ranking.txt", "a");

if (rank) {

fprintf(rank, "%s %d\n", nome, 6 - chances); // salva o nome do jogador vencedor e sua pontuação

fclose(rank);

} else {

printf("Erro ao salvar no ranking.\n"); // caso haja um erro, dará esse aviso

}

} else { // caso o jogador erre a palavra

forca(chances);

printf("Você perdeu! A palavra era: %s\n", palavra);

}

// libera a memória alocada da palavra escolhida

free(palavra);

free(palavra\_sec);

}

// função para exibir o nome do jogadores no ranking

void exibir\_ranking() {

FILE \*arquivo = fopen("ranking.txt", "r"); // abre o arquivo de ranking em modo leitura

if (!arquivo) { // verifica se o arquivo não existe ou não pôde ser aberto

printf("Nenhum ranking encontrado ou erro ao abrir o arquivo.\n");

return;

}

printf("\n=========== Ranking ===========\n");

char linha[100];

while (fgets(linha, sizeof(linha), arquivo)) {

printf("%s", linha); // exibe a linha na tela

}

printf("\n===============================\n");

fclose(arquivo);

}

// nome dos desenvolvedores

void exibir\_creditos() {

printf(">Créditos:\n");

printf("Guilherme Mendonça Silva\n");

printf("Zeus Fontes Barbosa\n");

printf("Brenno Lucas Sabino\n");

printf("Eduardo Costa Dias");

}

int main() {

setlocale(LC\_ALL, "Portuguese");

menu(); // chama o menu principal

return 0;

}