

Técnicas e Desenvolvimento de Algoritmo

Atividade avaliativa

Alunos: Ana Luiza Brandão Araújo (RGM: 39233545), Israel Garibaldi (RGM: 38968169),
Ana Beatriz Silva (RGM: 38640805) e Ana Beatriz Costa (RGM: 38488612)

Relatório do projeto de Técnicas e Desenvolvimento de Algoritmo

Introdução

No dia 11 de novembro de 2024, o professor universitário Wallace Bonfim, responsável por lecionar a matéria de Técnicas e Desenvolvimento de Algoritmo, propôs um projeto para a turma D do 2º período de Ciência da Computação.

O projeto consiste em, através do conteúdo das aulas lecionadas, elaborar um jogo na linguagem de programação C e os alunos deveriam formar grupos de no mínimo 4 e no máximo 5 pessoas para o devido trabalho. Além disso, o projeto é dividido em 3 segmentos: O algoritmo, o qual é a construção do código com os itens das técnicas e boas práticas, a organização da solução em funções lógicas bem definidas, o tratamento de erros, e o uso de recursos adequados da linguagem C para executar corretamente; o relatório, em que deverá conter a descrição geral do jogo e das regras com exemplificação de código fonte, dificuldades encontradas e soluções implementadas, demonstrativo das funcionalidades implementadas, descrevendo e registrando o algoritmo em funcionamento com prints de tela; e o vídeo demonstrativo de 2 a 5 minutos demonstrando a solução em funcionamento.

O docente estabeleceu um prazo de 2 semanas para enviar o projeto na plataforma de ensino BlackBoard, indo do dia 11 de novembro até dia 28 de novembro, com uma pontuação de até 2,5 pontos na média.

O jogo escolhido pelo grupo foi o jogo da forca, no qual o jogador tem que acertar qual é a palavra oculta, tendo como dica o número de letras e o tema ligado à palavra, e toda vez que uma letra errada é digitada, ou seja, que não se encontra na palavra, uma parte do corpo do enforcado é desenhado, então o jogo termina com o acerto da palavra ou a morte do enforcado.

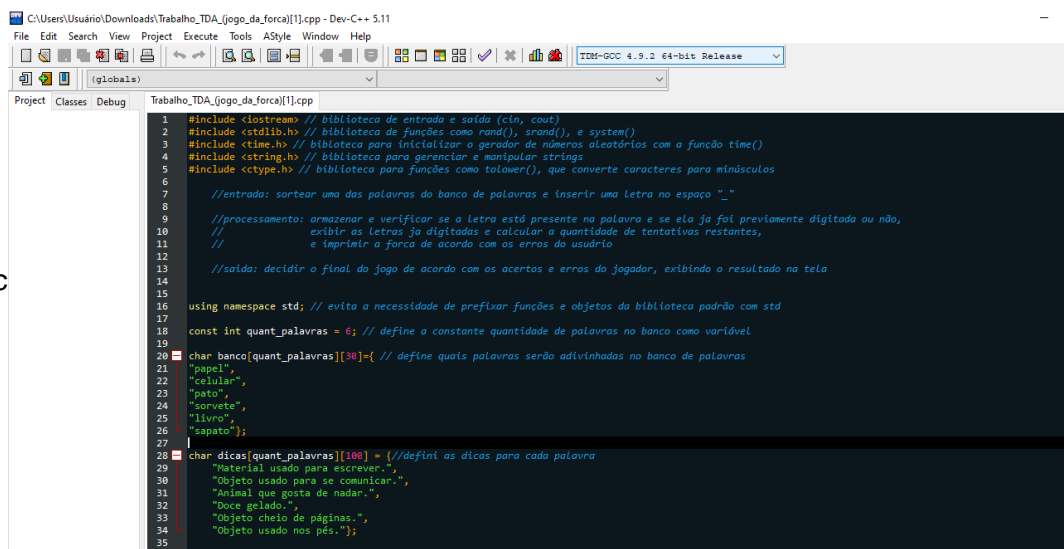
Resolução

O grupo começou a construir o projeto com a definição do jogo e decomposição das etapas, pois o jogo deveria ter: um banco de palavras, dicas para adivinhar as palavras, um sorteador de palavra do banco para ser utilizado na partida, funções para imprimir na tela a palavra digitada, copiar a palavra sorteada do banco para um array de palavras sorteadas, preencher a palavra digitada com _ para indicar que nenhuma letra foi descoberta, desenhar a forca, verificar se a letra já foi digitada anteriormente ou não, imprimir a letra correta na palavra oculta, exibir as letras já digitadas e verificar se a palavra oculta já foi completada.

Além disso, existiria uma função de menu com opções de jogar, créditos e sair, e o código fonte deveria ser organizado, estruturado e documentado, explicando cada bloco de funcionalidade do jogo.

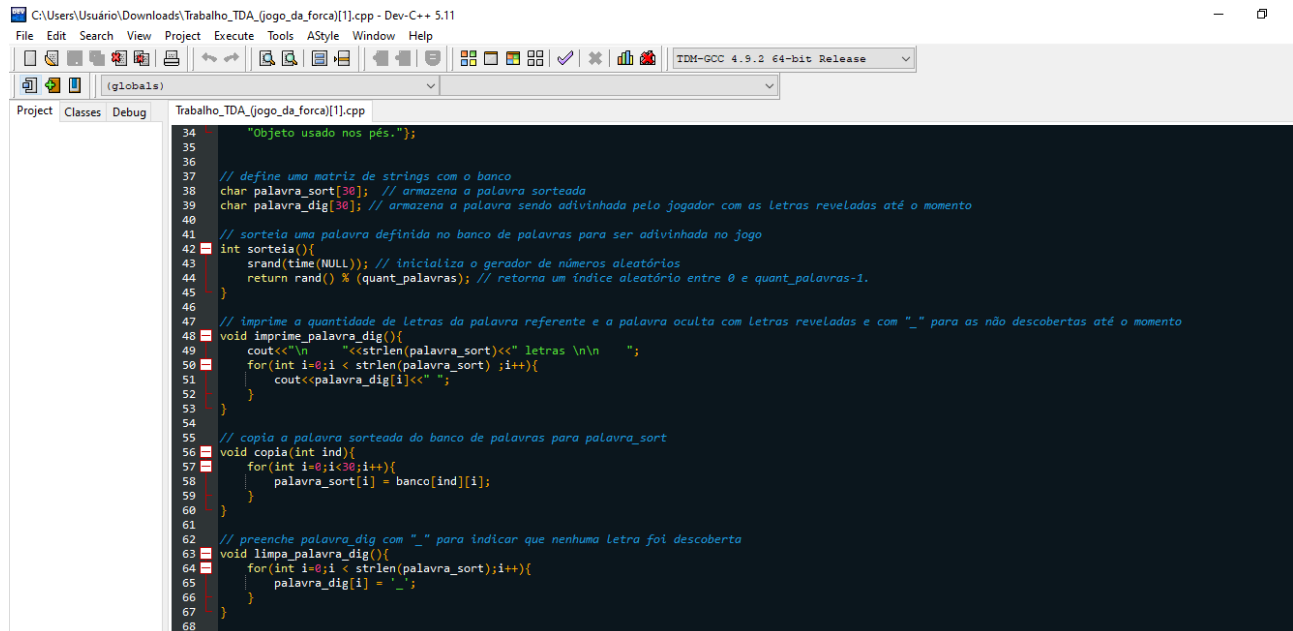
Após a definição e decomposição das etapas, o grupo escreveu as bibliotecas necessárias para o trabalho que são a biblioteca `#include <iostream>` para comandos de entrada e saída (`cin`, `cout`), a `#include <stdlib.h>` para funções como `rand()`, `srand()`, e `system()` e `#include <time.h>` para a função `time()`, essenciais principalmente para inicializar o sorteador, a `#include <string.h>` para gerenciar e manipular strings, e a `#include <ctype.h>` para a função `tolower()`, que converte caracteres para minúsculos. Dessa forma, as práticas boas foram estabelecidas com a entrada sendo sortear uma das palavras do banco de palavras e inserir uma letra no espaço "_", o processamento sendo armazenar e verificar se a letra está presente na palavra e se ela já foi previamente digitada ou não, exibir as letras já digitadas e calcular a quantidade de tentativas restantes, e imprimir a forca de acordo com os erros do usuário, e saída sendo decidir o final do jogo de acordo com os acertos e erros do jogador, exibindo o resultado na tela. Em seguida, foram escritos os bancos de palavras, contendo 6 palavras, e o de dicas, a matriz de strings para armazenar a palavra sorteada (`palavra_sort[30]`) e a palavra sendo adivinhada (`palavra_dig[30]`) pelo jogador com as letras reveladas até o momento. Com isso, o grupo partiu para a definição das principais funções.

Bloco
de
código
das
bibliotecas
e
dos
bancos
de
dados



```
1 #include <iostream> // biblioteca de entrada e saída (cin, cout)
2 #include <stdlib.h> // biblioteca de funções como rand(), srand(), e system()
3 #include <time.h> // biblioteca para inicializar o gerador de números aleatórios com a função time()
4 #include <string.h> // biblioteca para gerenciar e manipular strings
5 #include <ctype.h> // biblioteca para funções como tolower(), que converte caracteres para minúsculos
6
7 //entrada: sortear uma das palavras do banco de palavras e inserir uma letra no espaço "_"
8
9 //processamento: armazenar e verificar se a letra está presente na palavra e se ela já foi previamente digitada ou não,
10 //exibir as letras já digitadas e calcular a quantidade de tentativas restantes,
11 //e imprimir a forca de acordo com os erros do usuário
12
13 //saída: decidir o final do jogo de acordo com os acertos e erros do jogador, exibindo o resultado na tela
14
15
16 using namespace std; // evita a necessidade de prefixar funções e objetos da biblioteca padrão com std
17
18 const int quant_palavras = 6; // define a constante quantidade de palavras no banco como variável
19
20 char banco[quant_palavras][30] = { // define quais palavras serão adivinhadas no banco de palavras
21     "papel",
22     "celular",
23     "pato",
24     "sorvete",
25     "livro",
26     "sapato"
27 };
28
29 char dicas[quant_palavras][100] = { // defini as dicas para cada palavra
30     "Material usado para escrever.",
31     "Objeto usado para se comunicar.",
32     "Animal que gosta de nadar.",
33     "Doce gelado.",
34     "Objeto cheio de páginas.",
35     "Objeto usado nos pés."
36 };
```

Foram programadas a função `sorteia()`, em que a partir de um gerador de números aleatórios retorna um índice aleatório entre 0 e a quantidade de palavras do banco de dados; a `imprime_palavra_dig()`, na qual exibe a quantidade de letras da palavra referente e a palavra oculta com letras reveladas e com "_" para as não descobertas até o momento; a `copia()`, que somente copia a palavra sorteada do banco de palavras para `palavra_sort`; a `limpa_palavra_dig()`, que preenche `palavra_dig` com "_" para indicar que nenhuma letra foi descoberta.

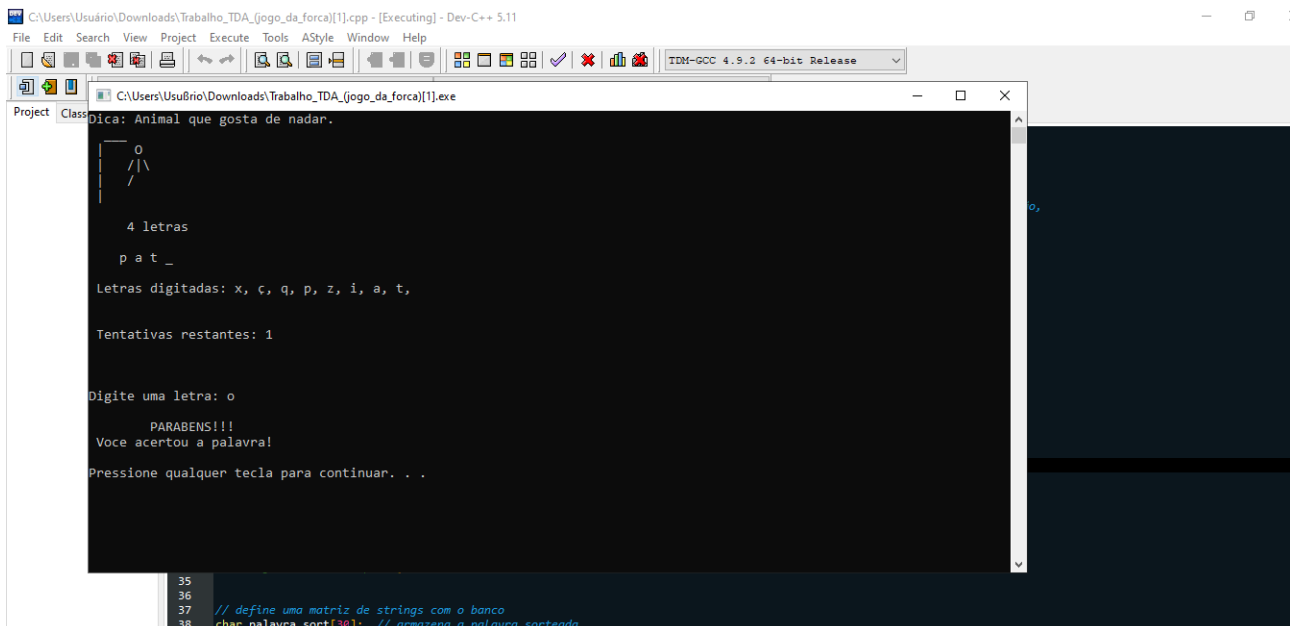


```
34 "Objeto usado nos pés.");
35
36 // define uma matriz de strings com o banco
37 char palavra_sort[30]; // armazena a palavra sorteada
38 char palavra_dig[30]; // armazena a palavra sendo adivinhada pelo jogador com as letras reveladas até o momento
39
40 // sorteia uma palavra definida no banco de palavras para ser adivinhada no jogo
41 int sorteia(){
42     srand(time(NULL)); // inicializa o gerador de números aleatórios
43     return rand() % (quant_palavras); // retorna um índice aleatório entre 0 e quant_palavras-1.
44 }
45
46 // imprime a quantidade de letras da palavra referente e a palavra oculta com letras reveladas e com "_" para as não descobertas até o momento
47 void imprime_palavra_dig(){
48     cout<<"\n" <<strlen(palavra_sort)<<" letras \n\n" <<endl;
49     for(int i=0; i < strlen(palavra_sort); i++){
50         cout<<palavra_dig[i]<<" ";
51     }
52 }
53
54 // copia a palavra sorteada do banco de palavras para palavra_sort
55 void copia(int ind){
56     for(int i=0; i<30; i++){
57         palavra_sort[i] = banco[ind][i];
58     }
59 }
60
61 // preenche palavra_dig com "_" para indicar que nenhuma letra foi descoberta
62 void limpa_palavra_dig(){
63     for(int i=0; i < strlen(palavra_sort); i++){
64         palavra_dig[i] = '_';
65     }
66 }
67
68 }
```

Bloco de código das funções

Logo após foram definidas a `imprime_forca()`, em que desenha a forca com switch case de acordo com a quantidade de erros (`let_erro`); a `verifica_letra()`, na qual verifica se a letra já foi digitada, ou seja, se retornar verdadeira, a letra já foi usada e se for falsa a letra ainda não foi usada; a `substitui()`, onde revela a letra (caso for a correta) na palavra para ser adivinhada; a `imprime_let_dig()`; em que as letras já digitadas anteriormente são exibidas; e a `palavra_completa()`, na qual verifica se a palavra já foi completada.

Com todas as funções cruciais escritas, o grupo programa a parte prática, onde o usuário jogará, com a definição da função `jogar()` e a inicialização das funções estruturadas feitas anteriormente, sorteando o índice de uma palavra, copiando a palavra sorteada para `palavra_sort`, inicializando a `palavra_dig`, o contador de letras, o vetor de letras, a variável de letra digitada, mostrando a dica relacionada a palavra. Sendo assim, com a existência da estrutura de repetição `while(cont_errores<6)`, o sistema é capaz de limpar a tela, exibir a dica sempre antes da forca, imprimir a forca, apresentar o progresso da palavra, mostrar letras usadas, apresentar um bloco de tentativas restantes, verificar se a letra já foi digitada na palavra ou não e se deverá ser revelada na tela ou não, e dois blocos para finalizar o jogo de acordo com os acertos e erros do jogador, exibindo o resultado.



```
o
/\
/

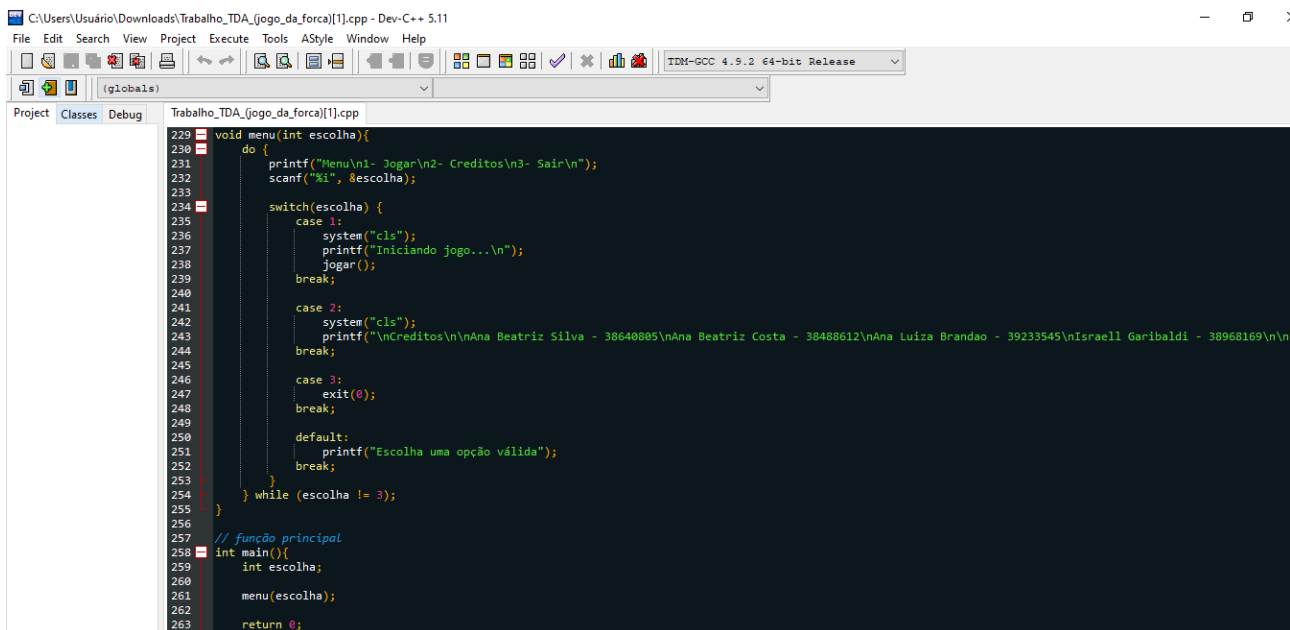
4 letras
p a t _
Letras digitadas: x, c, q, p, z, i, a, t,
Tentativas restantes: 1

Digite uma letra: o
PARABENS!!!
Voce acertou a palavra!
Pressione qualquer tecla para continuar. . .

35
36
37 // define uma matriz de strings com o banco
38 char palavra sort[30]; // armazena a palavra sorteada
```

Jogo executando no programa

Ao final do código, foi adicionado o menu, onde o usuário poderá escolher entre a opção de jogar, ver os créditos, onde estão os desenvolvedores do projeto, e sair do programa, com a estrutura condicional switch case, e a função principal para exibir o menu.



```
229 void menu(int escolha){
230 do {
231 printf("Menu\n1- Jogar\n2- Creditos\n3- Sair\n");
232 scanf("%i", &escolha);
233
234 switch(escolha) {
235 case 1:
236 system("cls");
237 printf("Iniciando jogo...\n");
238 jogar();
239 break;
240
241 case 2:
242 system("cls");
243 printf("\nCreditos\n\nAna Beatriz Silva - 38640805\nAna Beatriz Costa - 38488612\nAna Luiza Brandao - 39233545\nIsrael Garibaldi - 38968169\n\n");
244 break;
245
246 case 3:
247 exit(0);
248 break;
249
250 default:
251 printf("Escolha uma opção válida");
252 break;
253 }
254 } while (escolha != 3);
255 }
256
257 // função principal
258 int main(){
259 int escolha;
260
261 menu(escolha);
262
263 return 0;
264 }
```

Bloco de código do menu

Apesar de conseguir concluir o código e fazer com que ele execute bem, no início, o grupo enfrentou dificuldades na execução concreta da ideia do jogo na linguagem C, porém, após pesquisar aprofundadamente sobre a funcionalidade de cada linha de código, a equipe foi capaz de desenvolver um jogo da forca.

Apêndice A- Trabalho_TDA_(jogo_da_forca)

Neste trabalho, foi desenvolvido o jogo da forca na linguagem C, contendo toda a parte técnica em forma de blocos e funções de código estruturado e documentado na linguagem C.