**Relatório**

**Curso:** Ciência da Computação.

**Matérias:** Técnicas e Desenvolvimentos de Algoritmos (TDA) e Laboratório e Desenvolvimento de Algoritmos (LDA).

**Data:** 08/12/2021

**Assunto:** Trabalho em equipe com objetivo de criar um programa de jogo (jogo da velha)

**Interessado:** O projeto foi solicitado pelos professores Douglas Andrade de Meneses e Wallace Sartori Bonfim.

**Integrantes do grupo:**

Arthur Lacerda RGM: 25464329

Luiz Felipe Vilhena RGM: 25367528

Lailton Abdon RGM: 26690624

Lucas Montenegro RGM: 26275830

**Anexo:** cópia do código fonte do jogo.

**Introdução:**

O programa foi realizado em equipe, as dúvidas que surgiram durante o desenvolvimento do projeto foram dirimidas por meio vídeo conferência.

O objetivo do jogo é conseguir três círculos ou três xis em linha, quer horizontal, vertical ou diagonal, e ao mesmo tempo, quando possível, impedir o adversário de ganhar na próxima jogada. O tabuleiro é uma matriz de três linhas por três colunas, dois jogadores escolhem uma marcação cada um, geralmente um círculo (O) e um xis (X). A lógica do jogo é muito simples, de modo que não é difícil deduzir ou decorar todas as possibilidades para efetuar a melhor jogada – apesar de o número total de possibilidades ser muito grande, a maioria delas é simétrica, além de que as regras são simples.

Os competidores jogam alternadamente, uma marcação por vez, numa lacuna que esteja vazia, quando um deles conquista o objetivo, costuma-se riscar os três símbolos.

**Regras do Jogo:**

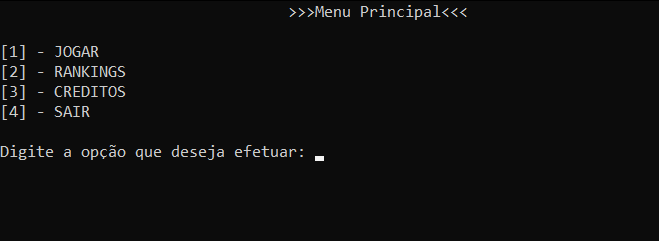
As regras são simples, por esse motivo, é muito comum que o jogo empate (ou “dê velha”).

1. Ganhar: Se você tem duas peças numa linha, ponha a terceira.
2. Bloquear: Se o oponente tiver duas peças em linha, ponha a terceira para bloqueá-lo.
3. Triângulo: Crie uma oportunidade em que você poderá ganhar de duas maneiras.
4. Bloquear o Triângulo do oponente
5. Centro: Jogue no centro.
6. Canto vazio: jogue num canto vazio.

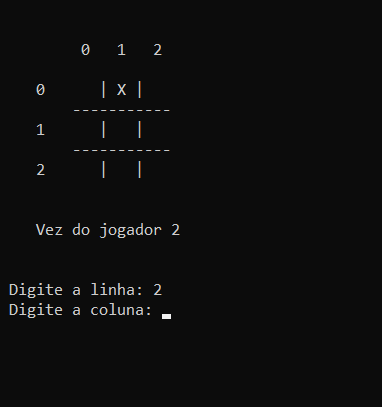
**Dificuldades e solução dos problemas:**

Durante o processo de desenvolvimento do projeto surgiram algumas dificuldades, porém com a colaboração de todos os integrantes e vários testes chegou-se a o produto final. As dificuldades mais recorrentes foram: passar o parâmetro nas funções; duvidas na hora de processar os dados; em qual estrutura usar if, else, while, for ou do while; falta de pratica e conhecimento quando se trata de arquivos, e muita dificuldade de aplicar o que estávamos pensando no código.

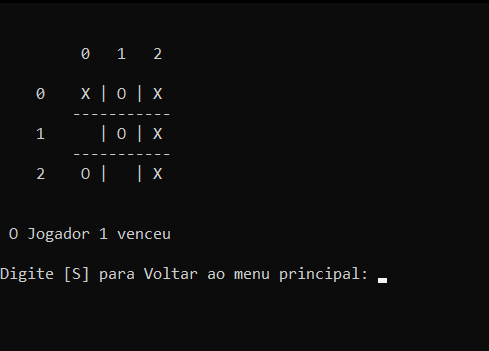
Algoritmo:



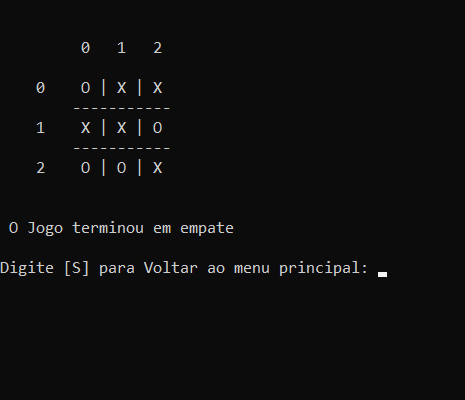
Aqui temos o menu do jogo com quatro opções disponíveis no qual você pode escolher a opção que lhe convém apenas digitando o valor correspondente da opção desejada.



Já aqui conseguimos ver um jogo já em execução no qual o jogador 1 marcou linha = 0 e coluna = 1, e o jogador 2 ainda não finalizou sua jogada.



Já nessa o jogo já foi finalizado tendo como resultado a vitória do jogador 1 já que ele tem a coluna 2 inteira preenchida com xis (X).



Aqui o jogo já foi finalizado só que dessa vez sem vitórias já que nenhum dos competidores conseguiram preencher nenhuma linha, ou coluna e nenhuma diagonal.

**Apêndice:**

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

char jogo[3][3];

int lin, col;

typedef struct{

int vitoria\_x;

int vitoria\_o;

int empate;

}Num\_Vitorias;

// Iniciando a matriz do jogo da velha com ' ' nas posições

void inicializar\_jogo(){

for(lin = 0; lin < 3; lin++){

for(col = 0; col < 3; col++){

jogo[lin][col] = ' ';

}

}

}

// Imprimindo o jogo da velha na tela

void imprimir\_jogo(){

printf("\n\n\t 0 1 2\n\n"); //colocando os índices colunas

for(lin = 0; lin < 3; lin++){

printf(" %d", lin); // colocando os índices linhas

for(col = 0; col < 3; col++){

if(col == 0) //Esta linha da um espaço no começo do jogo para organizar

printf("\t");

printf(" %c ", jogo[lin][col]);

if(col < 2)

printf("|"); // colocando "|" nas posições 0 e 1

}

printf("\n");

if(lin < 2)

printf("\t-----------\n"); // colocando uma linha nas posições 0 e 1

}

}

// Verificar vitória por linha

int ganhar\_por\_linha(int lin, char c){

if(jogo[lin][0] == c && jogo[lin][1] == c && jogo[lin][2] == c)

return 1;

else

return 0;

}

int ganhar\_por\_linhas(char c){

int ganhar = 0;

for(lin = 0; lin < 3; lin++){

ganhar += ganhar\_por\_linha(lin, c);

}

return ganhar;

}

// Verificar vitória por coluna

int ganhar\_por\_coluna(int col, char c){

if(jogo[0][col] == c && jogo[1][col] == c && jogo[2][col] == c)

return 1;

else

return 0;

}

int ganhar\_por\_colunas(char c){

int ganhar = 0;

for(col = 0; col < 3; col++){

ganhar += ganhar\_por\_coluna(col, c);

}

return ganhar;

}

// Verificar vitória na diagonal principal

int ganhar\_na\_diagonal\_p(char c){

if(jogo[0][0] == c && jogo[1][1] == c && jogo[2][2] == c)

return 1;

else

return 0;

}

// Verificar vitória na diagonal secundaria

int ganhar\_na\_diagonal\_s(char c){

if(jogo[0][2] == c && jogo[1][1] == c && jogo[2][0] == c)

return 1;

else

return 0;

}

// Verificar se a coordenada é válida

int verificar\_coordenada(int lin, int col){

if(lin >= 0 && lin < 3 && col >= 0 && col < 3 && jogo[lin][col] == ' ')

return 1;

else

return 0;

}

void ler\_coordenadas(char c){

int linha, coluna;

printf("\n\n");

printf(" Digite a linha: ");

scanf("%d", &linha);

printf(" Digite a coluna: ");

scanf("%d", &coluna);

printf("\n");

while(verificar\_coordenada(linha, coluna) == 0){

printf(" Digite uma coordenada válida para a linha: ");

scanf("%d", &linha);

printf(" Digite uma coordenada válida para a coluna: ");

scanf("%d", &coluna);

}

jogo[linha][coluna] = c;

}

// Verificar empate

int verificar\_empate(){

int empate = 0;

for(lin = 0; lin < 3; lin++){

for(col = 0; col < 3; col++){

if(jogo[lin][col] == ' '){

if(ganhar\_por\_linhas('X') == 1 || ganhar\_por\_colunas('X') == 1 || ganhar\_na\_diagonal\_p('X') == 1 || ganhar\_na\_diagonal\_s('X') == 1){

empate = empate + 0;

}

else if(ganhar\_por\_linhas('O') == 1 || ganhar\_por\_colunas('O') == 1 || ganhar\_na\_diagonal\_p('O') == 1 || ganhar\_na\_diagonal\_s('O') == 1){

empate = empate + 0;

}

else{

empate++;

}

}

}

}

return empate;

}

void jogar(){

int rodada = 1, vitoria\_X = 0, vitoria\_O = 0;

do{

imprimir\_jogo();

if (rodada == 1){

printf("\n\n Vez do jogador 1\n");

ler\_coordenadas('X');

rodada++;

vitoria\_X += ganhar\_por\_linhas('X');

vitoria\_X += ganhar\_por\_colunas('X');

vitoria\_X += ganhar\_na\_diagonal\_p('X');

vitoria\_X += ganhar\_na\_diagonal\_s('X');

}

else{

printf("\n\n Vez do jogador 2\n");

ler\_coordenadas('O');

rodada = 1;

vitoria\_O += ganhar\_por\_linhas('O');

vitoria\_O += ganhar\_por\_colunas('O');

vitoria\_O += ganhar\_na\_diagonal\_p('O');

vitoria\_O += ganhar\_na\_diagonal\_s('O');

}

system("cls");

}while(vitoria\_X == 0 && vitoria\_O == 0 && verificar\_empate() > 0);

imprimir\_jogo();

if(verificar\_empate() == 0 && vitoria\_X == 0 && vitoria\_O == 0)

printf("\n\n O Jogo terminou em empate");

else if(vitoria\_O == 1)

printf("\n\n O Jogador 2 venceu");

else

printf("\n\n O Jogador 1 venceu");

}

void menu\_jogo(){

inicializar\_jogo();

fflush(stdin);

int opcao;

system("cls");

printf("\t\t\t\t>>>Menu Principal<<<\t\t\t\t\n\n");

printf("[1] - JOGAR \n[2] - RANKINGS \n[3] - CRÉDITOS \n[4] - SAIR\n\nDigite a opção que deseja efetuar: ");

opcao = getchar();

switch(opcao){

case '1':

system("cls");

jogar();

break;

case '2':

system("cls");

printf("\t\t\t\t>>>Ranking<<<\n\n\n");

FILE \*arq;

char Linha[100];

char \*result;

int i;

arq = fopen("ranking.txt", "rt");

for (i = 1; i < 4; i++){

result = fgets(Linha, 100, arq);

if(result){

if(i < 3)

printf("O número de vitórias do jogador [%d] é: %s", i, Linha);

if(i == 3){

printf("\nO Número de empates é: %s", Linha);

break;

}

}

}

fclose(arq);

break;

case '3':

system("cls");

printf("\t\t\t\t>>>Créditos<<<\n\n");

printf("Arthur Lacerda\n");

printf("Lucas Montenegro Nunes\n");

printf("Luiz Felipe Vilhena\n");

printf("Lailton Abdon\n");

break;

case '4':

system("cls");

FILE \*limpar;

limpar = fopen("ranking.txt", "w");

fclose(limpar);

exit(0);

default:

printf("\nVocê deve escolher uma opção válida\n");

printf("Pressione qualquer tecla para voltar ao menu\n");

system("pause");

break;

}

}

int main(){

setlocale(0, "Portuguese");

char op;

FILE \*pont\_arq;

Num\_Vitorias result;

pont\_arq = fopen("ranking.txt", "w+b");

fprintf(pont\_arq, "%d\n%d\n%d", result.vitoria\_x = 0, result.vitoria\_o = 0, result.empate = 0);

fclose(pont\_arq);

while(1){

menu\_jogo();

if(ganhar\_por\_linhas('X') == 1 || ganhar\_por\_colunas('X') == 1 || ganhar\_na\_diagonal\_p('X') == 1 || ganhar\_na\_diagonal\_s('X') == 1){

result.vitoria\_x++;

}

if(ganhar\_por\_linhas('O') == 1 || ganhar\_por\_colunas('O') == 1 || ganhar\_na\_diagonal\_p('O') == 1 || ganhar\_na\_diagonal\_s('O') == 1){

result.vitoria\_o++;

}

if(verificar\_empate() == 0){

if(ganhar\_por\_linhas('X') == 1 || ganhar\_por\_colunas('X') == 1 || ganhar\_na\_diagonal\_p('X') == 1 || ganhar\_na\_diagonal\_s('X') == 1){

result.empate = result.empate + 0;

}

else if(ganhar\_por\_linhas('O') == 1 || ganhar\_por\_colunas('O') == 1 || ganhar\_na\_diagonal\_p('O') == 1 || ganhar\_na\_diagonal\_s('O') == 1){

result.empate = result.empate + 0;

}

else{

result.empate++;

}

}

fflush(stdin);

pont\_arq = fopen("ranking.txt", "r+b");

fprintf(pont\_arq, "%d\n%d\n%d", result.vitoria\_x, result.vitoria\_o, result.empate);

fclose(pont\_arq);

printf("\n\nDigite [S] para voltar ao menu principal: ");

op = getchar();

if(op != 'S' && op != 's'){

exit(0);

}

}

return 0;

}